Towards Formal Structural Representation of Spoken Language: An Evolving Transformation System (ETS) Approach

Alexander Gutkin

Thesis submitted for the degree of Doctor of Philosophy

School of Informatics
University of Edinburgh

2005

Abstract

Speech recognition has been a very active area of research over the past twenty years. Despite an evident progress, it is generally agreed by the practitioners of the field that performance of the current speech recognition systems is rather suboptimal and new approaches are needed. The motivation behind the undertaken research is an observation that the notion of representation of objects and concepts that once was considered to be central in the early days of pattern recognition, has been largely marginalised by the advent of statistical approaches. As a consequence of a predominantly statistical approach to speech recognition problem, due to the numeric, feature vector-based, nature of representation, the classes inductively discovered from real data using decision-theoretic techniques have little meaning outside the statistical framework. This is because decision surfaces or probability distributions are difficult to analyse linguistically. Because of the later limitation it is doubtful that the gap between speech recognition and linguistic research can be bridged by the numeric representations. This thesis investigates an alternative, structural, approach to spoken language representation and categorisation. The approach pursued in this thesis is based on a consistent program, known as the Evolving Transformation System (ETS), motivated by the development and clarification of the concept of structural representation in pattern recognition and artificial intelligence from both theoretical and applied points of view.

This thesis consists of two parts. In the first part of this thesis, a similarity-based approach to structural representation of speech is presented. First, a linguistically well-motivated structural representation of phones based on distinctive phonological features recovered from speech is proposed. The representation consists of string templates representing phones together with a similarity measure. The set of phonological templates together with a similarity measure defines a symbolic metric space. Representation and ETS-inspired categorisation in the symbolic metric spaces corresponding to the phonological structural representation are then investigated by constructing appropriate symbolic space classifiers and evaluating them on a standard corpus of read speech. In addition, similarity-based isometric transition from phonological symbolic metric spaces to the corresponding non-Euclidean vector spaces is investigated.

Second part of this thesis deals with the *formal* approach to structural representation of spoken language. Unlike the approach adopted in the first part of this thesis, the representation developed in the second part is based on the mathematical language of the ETS formalism. This formalism has been specifically developed for structural modelling of dynamic processes. In particular, it allows the representation of both objects and classes in a uniform event-based hierarchical framework. In this thesis, the latter property of the formalism allows the adoption of a more physiologically-concrete

approach to structural representation. The proposed representation is based on gestural structures and encapsulates speech processes at the articulatory level. Algorithms for deriving the articulatory structures from the data are presented and evaluated.

Acknowledgements

First and foremost the author would like to thank Lev Goldfarb, a friend and mentor, whose profound vision has inspired and supported me during all these years. Utmost thanks to my supervisor Simon King, whose patience, support and expert advice allowed this thesis to come to fruition. Many thanks to my colleagues at the Centre for Speech Technology Research (CSTR), Institute for Communicating and Collaborative Systems (ICCS) and Department of Linguistics, whose continuous encouragement and support have been instrumental in completing this work: Rob Clark, Joe Frankel, Steve Isaard, Caroline Hastings, Christine Haunz, Avril Heron, Ari Krupnikov, Korin Richmond, Miles Osborne, Steve Renals, Marc Steedman, Virve-Anneli Vihman and Mirjam Wester. Continuous encouragement and support from David Gay, Oleg Golubitsky, Dmitry Korkin, John Abela and other colleagues, past and present, from Inductive Informatics Group at University of New Brunswick, without whom this work would not be possible, are gratefully acknowledged. Additional thanks to Robert Duin, Alfons Juan Ciscar, Grigory Gimmelfarb, David Mountain and Rémy Pujol for technical help and stimulating discussions.

Special thanks to Paul Taylor and Rhetorical Systems, Ltd. (now sadly defunct) for providing generous funding during all these years and to Centre for Speech Technology Research for supporting me during the last year of my work. The author would like to thank his Rhetorical colleagues, especially the following people: Mathew Aylett, Paul Chorley, Keith Edwards, Ann Devitt, Justin Fackrell, Janet Forbes, Katrin Hammervold, Ian Hodson, David McKelvie, Laurence Molloy, Liam Parker, Peter Rutten, Tony Robison, Wojciech Skut, David Talkin, Anthony Tomlinson, David Toney and Stefan Ulrich.

My deepest gratitude goes to my parents and Inez, for putting up with me all these years and, finally, to my very special friends Valery Barer, Eduard Bersudsky, Sarah Gingell, Kirill Gokhberg, Roman Goldenberg, Olga Goubanova, Rodion Goubanov, Pál Hegedüs, Tõnis Kriisa, Valery Levental, Zeev Lieber, Jelena Meznaric, Kathrin Schödel, Yulia Shtutina and Tatyana Jakovskaya for being there for me.

Всем огромное спасибо!

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification.

(*Alexander Gutkin*)

Моим родителям и Инес посвящается

Contents

1 Introduction **1**

1.1 Pattern Recognition: A Brief Overview 2

1.1.1 The Fundamental Tasks of Pattern Recognition 3

1.1.2 Representations in Pattern Recognition 4

1.1.3 Numeric Class Representation and Generalisation 6

1.1.4 Structural Class Representation and Generalisation 11

1.2 Representations in Speech Modelling and Recognition 15

1.2.1 Speech Recognition Problem: An Overview 15

1.2.2 Numeric Representations 25

1.2.3 Structural Representations 27

1.3 Research Motivations 30

1.3.1 Current Situation with Representations 30

1.3.2 Unification of Structural and Numeric Approaches 34

1.3.3 Topological Class-Centric Approach to Speech Representation . . 36

1.3.4 Formal Approach to Speech Representation 37

1.3.5 Brief ETS Literature Overview 40

1.4 Research Objectives 43

1.5 Thesis Organisation 44

1.6 Publications and Declaration 46

I Topological Approaches to Structural Representation **47**

2 Phonological Symbolic Metric Spaces **48**

2.1 Introduction 48

2.2 Preliminaries: Metric Spaces 50

2.3 Phonological Object Representation 51

2.3.1 Atomic Representational Units: Phonological Features 51

2.3.2 Detecting Distinctive Features in Continuous Speech 53

2.3.3 Transition to Symbolic Space 55
2.3.4 Phonological Templates 57
2.4 Phonological Metrics and Metric Space 59
2.4.1 Phonological Metric Space 59
2.4.2 String Metrics 60
2.5 Prototype Selection and Classification 65
2.5.1 Template Means and Medians 66
2.5.2 Clustering Algorithms 67
2.5.3 Clustering Initialisation Criteria 68
2.5.4 Classification 69
2.6 Experiments and Discussion 70
2.6.1 The Database 70
2.6.2 Multivalued Feature Detection 71
2.6.3 Derivation of Phonological Templates 72
2.6.4 Training Set Pruning 73
2.6.5 Classification 74
2.7 Summary and Potential Improvements 75

3 Pseudo-Euclidean Embedding of Phonological Metric Spaces 80
3.1 Introduction 80
3.2 Preliminaries: Pseudo-Euclidean Vector Spaces 81
3.2.1 Symmetric Bilinear Forms 82
3.2.2 Pseudo-Euclidean Space 85
3.3 From Metric to Pseudo-Euclidean Space: Isometric Embeddings 87
3.3.1 Linear Embedding 89
3.3.2 Dimensionality Reduction 95
3.4 Projection of Unseen Objects 98
3.4.1 Basic Metric Projection 99
3.4.2 Corrected Metric Projection 101
3.4.3 Selecting the Basis for Metric Projections 103
3.5 Experiments and Discussion 105
3.5.1 Three-class Problem 106
3.5.2 Full Problem 114
3.6 Summary and Potential Improvements 117

4 Inductive Learning with $\mathbf{ETS}_0$ 120
4.1 Introduction 120
4.2 Preliminaries: Objects, Transformations and Metrics 124

4.2.1 Structural Dissimilarity Measures . . . 125
4.2.2 Block Edit Distances on Strings . . . 127
4.3 Evolving Transformation Systems (ETS_0) Model . . . 130
4.3.1 Transformation System . . . 130
4.3.2 Evolving Transformation System . . . 136
4.3.3 Learning in Evolving Transformation System . . . 138
4.3.4 Inductive Class Representation . . . 144
4.4 Experiments and Discussion . . . 146
4.4.1 Three-class Problem . . . 147
4.4.2 Full Problem . . . 151
4.5 Summary and Potential Improvements . . . 154

II Structural Representation Formalism 158

5 Formal Articulatory Representation of Speech with ETS_2 159
5.1 Introduction . . . 159
5.2 Preliminaries: The ETS_2 Representation Formalism . . . 162
5.2.1 Primitive Transformations . . . 162
5.2.2 Instances of Structural History . . . 165
5.2.3 Extructs . . . 168
5.2.4 Transformations and Supertransformations . . . 170
5.2.5 Level Ascension Postulate . . . 173
5.2.6 Inductive Structure . . . 174
5.3 Articulatory Representation . . . 178
5.3.1 Primitive Articulatory Gesture . . . 179
5.3.2 The Articulatory Corpus . . . 181
5.3.3 Primitive Gestures and Their Groups . . . 182
5.3.4 Automatic Detection of Primitive Gestures . . . 184
5.3.5 Gestural Formations as ETS_2 Structs . . . 186
5.3.6 Articulatory Transformations . . . 188
5.3.7 Class Description via ETS_2 Transforms . . . 190
5.3.8 Matching Gestural Transformations . . . 194
5.3.9 Higher Levels of Representation . . . 196
5.4 Experiments and Discussion . . . 198
5.4.1 Gesture Detection . . . 198
5.4.2 Phoneme Classification . . . 201
5.5 Summary and Potential Improvements . . . 203

6 Conclusions and Future Research **208**

6.1 Thesis Summary and Results 208

6.1.1 Topological Approach 208

6.1.2 Formal Approach 210

6.2 Contributions of this Thesis 211

6.3 Open Issues 212

6.3.1 Topological Approach 212

6.3.2 Formal Approach 214

6.4 Future Work 215

6.4.1 Articulatory Representation 215

6.4.2 Speech Recognition Problem Revisited 216

6.5 Concluding Remark 216

Bibliography **218**

List of Tables

2.1 Multivalued feature system 53
2.2 Neural network architectures 71
2.3 Statistics for training and test sets 72
2.4 Concepts and algorithms for clustering 73
2.5 Phoneme classification results 74

3.1 Phoneme classification results for the three class problem 113

4.1 Classification results for a three-class task 149
4.2 Statistics on discovered non-trivial ETS_0 transformations 153
4.3 Classification results for a full problem 153

5.1 Critical articulators 182
5.2 Groups of primitive gestures 183
5.3 Phonemes and constituent gestures 194
5.4 Critical primitive gestures 200
5.5 Evaluation results 201
5.6 Phoneme classification results 204

List of Figures

1.1 An example of monophone HMM 18
1.2 A simple word-level HMM model 18
1.3 Partial transition structure of recognition network 21
1.4 Hierarchical organisation of speech knowledge 22
1.5 Formative history 39
1.6 A two-level ETS_4 representation 41

2.1 Output of a neural network feature detector 55
2.2 Phonemic and syllabic boundaries of a sample utterance 56
2.3 Quantisation of neural network outputs 57
2.4 Matrix representation of a phonological template 58
2.5 Example of a phonological template 58
2.6 Normalised Edit Distance algorithm 65
2.7 Approximate Mean String algorithm 77
2.8 *MaxMin* clustering initialisation algorithm 78
2.9 k-NN AESA algorithm 79

3.1 Example pseudo-Euclidean spaces 86
3.2 Example of a four-dimensional metric space 89
3.3 An isometric embedding 90
3.4 Minimal isometric embedding 96
3.5 Orthogonal projection 97
3.6 Example of a reduced vector representation 98
3.7 Basic metric projection 100
3.8 Corrected metric projection 102
3.9 Example of a corrected metric projection 107
3.10 Dimensionality reduction for the three class problem 109
3.11 Dimensionality reduction for the full problem 115
3.12 Phoneme classification results for the full problem 117

4.1 Stream-specific phonological transformation systems 133
4.2 Learning within ETS_0 135
4.3 A two-dimensional unit simplex 135
4.4 A two-dimensional unit simplex in three dimensional parametric space . 138
4.5 Learning algorithm for ETS_0 139
4.6 Per-stream feature transformations 145
4.7 Error measures for a three-class task 150
4.8 Distance measures for a three-class task 152

5.1 Original primitives 164
5.2 Concrete primitives 165
5.3 Class primitives 165
5.4 Instances of structural history (structs) 167
5.5 Structs and their composition 168
5.6 Struct and the corresponding attachment graph 169
5.7 Example structs 170
5.8 Transformations 171
5.9 Supertransform 172
5.10 Supertransform and next-level original primitive 174
5.11 Multi-level inductive structure 176
5.12 Pyramid view of a supertransform 177
5.13 Pictorial view of an abstract gestural structure. 180
5.14 Groups of primitive gestures 184
5.15 EPG regions and consonantal stable phases 185
5.16 Tongue dorsum height gestures and EMA stream 187
5.17 Gestural ETS_2 struct 189
5.18 Common gestural patterns 191
5.19 Supertransform for phoneme [g] 193
5.20 Partial order specification for transform 195
5.21 ETS_2 Transform Matching Algorithm. 196
5.22 Next-level gestural primitives 197

Chapter 1

Introduction

The last two decades have seen significant advances in human-machine interfaces. Speech and language technology (speech recognition, in particular) is among several areas which have benefited enormously from these advances (Young, 2001). Numerous prototype systems developed within the research community are now extensively used both in commercial environments and are available to end-users (Greenberg, 2001). Among many reasons given for these advances (like technological improvements in hardware which led to the increase in the computational resources available for modelling), one of the most crucial factors to us seems to be the following: these impressive advances were made possible by the introduction of formally rigorous modelling framework (based on the advances in mathematical statistics, as we shall see in Section 1.2). This framework, on the one hand, is flexible enough to accommodate the variety of research models (incorporating recent results from the field of machine learning) and, on the other, robust and computationally efficient enough to allow for extensive experimentation and development of practical systems for commercial use.

Despite the evident progress in speech recognition, many researchers have argued that the performance of the state-of-the-art models, which emerged in the last two decades of the twentieth century, have reached a "local optimum" (the original goal was the "global optimum" defined as machine performance indistinguishable from human performance on natural speech) (Bourlard *et al.*, 1996; Deng, 1998; Deng *et al.*, 1997; Ostendorf, 1999; Young, 2001). It was argued that in order to rectify this situation, the current approaches to speech recognition need tighter integration with the methods and theories elaborated over the years by the linguistic community. Sadly enough, these results were often neglected. In recent years, alternative approaches to speech recognition, which are the result of a more careful development, have been gradually crystallising (e.g. Bilmes, 2003; Deng, 1998; Glass, 2003; King *et al.*, 2000; Livescu *et al.*, 2003) and are very promising. Together with the researchers who work on alternative approaches

to speech modelling and recognition, the author believes that new approaches are definitely desirable. The approach pursued in this thesis is the result of a consistent program aimed at research into formal approaches to *structural representation* of speech.

Although the traditional means of studying speech phenomena in linguistics have been symbolic (structural), the approaches to pattern representation in speech recognition are predominantly numeric. The alternative, structural, means of pattern representation have, however, received little attention. In our view, one of the main reasons for this situation is the apparent lack of suitable structural frameworks possessing the necessary formal power to accommodate the *class* representation of complex linguistic phenomena (e.g. phonemes and syllables). Sadly enough, this state of affairs also appears to apply to many other areas of pattern recognition (Pavlidis, 2003). It is hypothesised that the appearance of such a systematic analytical framework and the development of appropriate representations within it could potentially help in bridging the gap between, in particular, speech recognition and linguistic research. What is the basis for such a hypothesis? The concept of class is absolutely pervasive in many areas of science, including linguistics. Hence, it is reasonable to assume that spoken language modelling and recognition can benefit from models that allow the *derivation* of linguistically sound class representations from the data.

Before proceeding with the exposition into the proposed approach, in Section 1.1 we open with the brief description of a more general research area — pattern recognition. The reason for doing this is simple. On the one hand, modern speech recognition (among with other specialised fields, like optical character recognition) is partially subsumed by the field of pattern recognition. On the other hand, some of the fundamental theoretical ideas that appeared in the general pattern recognition literature have added to the motivation of research presented in this thesis. In Section 1.2, we overview several approaches to speech modelling practiced in modern day speech recognition. A special emphasis is placed on numeric and structural approaches to representation in speech recognition, since these approaches generally reflect the current situation in pattern recognition. Motivations behind the research undertaken in this thesis are described in Section 1.3. The major research objectives are set in Section 1.4. This chapter concludes with a description of the thesis structure (Section 1.5) and a list of current publications which have resulted from work on this thesis (Section 1.6).

1.1 Pattern Recognition: A Brief Overview

Pattern recognition refers to the ability of humans to perceive regularities in the observations in some environment. A *pattern* is generally viewed as an object which belongs

to some *class*. A *class*, in turn, is seen as a concept of a collection of objects. According to Watanabe (1985), the latter definition is, generally, in agreement with Platonic philosophy, where all similar objects are but the imperfect material realisations of an "ideal" object (concept), which exists in some alternative reality and cannot be directly comprehended by our senses. In reality, the humans do not need an "ideal" object to form an idea about classes. In order to form an idea about some class in a human mind, it actually suffices to be presented with a limited set of related objects which represent that class (this is in line with Aristotelian philosophy according to Watanabe, 1985, Section 4.4). In general, pattern recognition refers to the latter process: having been shown a few positive samples (objects "belonging" to a class), and perhaps a few negative ones (objects from a different class), one is able to tell if a new object (whose exact classification is unknown) belongs to this class. In this sense, pattern recognition forms the foundation of categorisation in humans (where the latter is understood as formation of the categorised knowledge). Moreover, it has been argued by some scientists that the mechanisms of perception (especially *inductive inference*, which is inferring a generality from a few concrete classes) and pattern recognition are, in fact, identical (e.g. Goertzel, 1993, Chapter 9). The latter view was succinctly summarised in 1956 by Schrödinger (2003, p. 96):

> "A single experience that is never to repeat itself is biologically irrelevant. Biological value lies only in learning the suitable reaction to a situation that offers itself again and again, in many cases periodically, and always requires the same response..."

1.1.1 The Fundamental Tasks of Pattern Recognition

The fundamental tasks encompassed by any pattern recognition system are the following (Duin and Pękalska, 2005; Duin *et al.*, 2004; Goldfarb, 2004):

Representation The first issue which needs to be addressed when modelling some real world phenomena (objects, processes and so on) is the issue of *representation*. The framework used for representation imposes certain (formal) restrictions on the form of the objects being modelled, hence representations in pattern recognition are only a simplified approximation of the corresponding phenomena. The degree of "faithfulness" of the representation depends, to a large extent, on the modelling power of the corresponding formal framework. An additional consideration is the incorporation of *a priori* domain-specific knowledge into the representation. Fundamental issues involved in the representation are treated in more detail in Section 1.1.2. Once the representation of the objects is obtained, the next step is generalisation.

Generalisation Traditionally, *generalisation* received the most attention in pattern recognition research. This is entirely justified by the absolutely crucial role played by this step in human perception. Generalisation encompasses two fundamentally related tasks: *learning* and *recognition*. Learning is commonly understood as an *inductive* process of constructing a representation of a set of classes based on a limited training set of objects representing these classes. Recognition, on the other hand, is, in theory[1], a *deductive* process in which the previously unseen objects are related to the representation of the concepts of the classes derived during the learning stage, rather than to the representation of the objects in the training set. Hence, it is important to make a distinction between the representation of objects and representation of classes. Several important points on the relation between the generalisation and the classes are presented in Section 1.1.3 (for statistical pattern recognition) and Section 1.1.4 (for structural pattern recognition).

1.1.2 Representations in Pattern Recognition

As mentioned earlier, the representation of some phenomenon is inextricably linked to the formal properties of the corresponding modelling framework (space). In modern pattern recognition, one usually distinguishes between the two approaches to modelling: *numeric* and *symbolic*[2]. The latter two types of representation delineate the two rather broad, and historically not very related, approaches to pattern recognition.

1.1.2.1 Numeric Representations

Numeric representations lie at the foundation of *statistical* pattern recognition (Duda *et al.*, 2001; Mitchel, 1997). The adjective "statistical" refers to the fact that this area of pattern recognition was motivated by (and, in fact, was derived from) mathematical statistics. In statistical pattern recognition, the representational spaces correspond to *vector spaces*. Historically, the overwhelming majority of the state-of-the-art approaches to statistical pattern recognition employ Euclidean vector spaces for modelling. A numeric representation of an object in a statistical pattern recognition framework is essentially an embedding of the data into a Euclidean space. In other words, the object is represented (encoded) as a *feature vector*. This process is sometimes called *feature selection*.

[1] In practice, it is sometimes possible to directly relate the previously unseen objects to the classes based on the training set objects.

[2] From this point onwards we will often refer to symbolic pattern recognition as *structural* pattern recognition. This is not to be confused with the structural *models* often employed in *statistical* (i.e. numeric) pattern recognition, such as graphical models (Zweig *et al.*, 2002).

It is generally agreed in statistical pattern recognition that the choice of features is a highly non-trivial task and no universally accepted procedures for feature selection exist. One of the main reasons for this is because the encoding of the objects into vectors requires domain-specific knowledge. In some areas of pattern recognition, the nature of the "object" is static, in others dynamic. In the latter case, the object refers to some dynamic process. Dynamic processes can only be approximated by sampling, which allows representation of the process by the sequence of feature vectors, each corresponding to an *instantaneous* (snapshot) representation of a process. In online handwriting recognition, for instance, one often encodes both the spatial (local geometry) and temporal (movement of a hand) information contained in strokes (Lui *et al.*, 2003). The letters and words are then encoded as sequences of the above vectors.

1.1.2.2 Structural Representations

The non-numeric, structural, representations are the subject of study within the area of *structural* pattern recognition (Bunke and Sanfeliu, 1990; Fu, 1982). Structural representations are more advanced than their numeric counterparts in terms of the modelling spaces they offer. Unlike numeric representations, which are limited to vector spaces, structural representations can be based on a wide variety of discrete algebraic structures — strings, trees, graphs and so on. They are better suited to model the *morphological makeup* of the corresponding objects and events. A very informal, but intuitive, observation will perhaps help to clarify this point: structural pattern recognition replaces a rigid encoding in terms of vector (or sequence thereof) with a discrete structure of an arbitrarily chosen complexity that preserves the "original" part/whole relationship describing the inter-dependencies between atomic constituents of an object.

It is not surprising that structural representations in pattern recognition predate the numeric ones because from the early days of pattern recognition it was assumed that there is more to the objects and events than just numeric features. In his book, Watanabe (1985) mentions the earliest (and surprisingly elegant) structural approaches: the cursive handwriting representation by Eden (1962) and the chromosome representation by Ledley *et al.* (1966). Among the most popular modern structural representations are string-based representations of molecular sequences in computational biology (Gusfield, 1997). We conclude by noting that the structural representations require an even greater domain-specific knowledge to properly encode the structural richness of the corresponding phenomenon.

1.1.3 Numeric Class Representation and Generalisation

Statistical approaches to pattern recognition use decision-theoretic methods in a feature vector space. Decision-theoretic methods were extensively studied in the early days of cybernetics, pattern recognition and mathematical statistics. The numerous theoretical and practical advances in these fields led to the emergence of several overlapping fields of activity — computational learning theory (Devroye *et al.*, 1996; Vapnik, 1998), machine learning (Mitchel, 1997) and neural networks (Bishop, 1995) among the others. See Kulkarni and Lugosi (1998) for a rather involved, but excellent, overview presented from the point of view of computational learning theory.

In statistical approaches, the objects and events in the environment are usually encoded as real-valued feature vectors. Let d be the dimension of the feature vector $x \in \mathbb{R}^d$. In this case, the representation of a real world object engendered by x is a point in a d-dimensional vector space. Informally, each such vector can be seen as a result of d instantaneous observations of the corresponding object in the environment. When defining generalisation, we mentioned that it is essentially a two stage procedure consisting of learning and recognition steps. The goal of the learning (or *training*) stage is to infer a representation of classes. In statistical language, this is equivalent to determining the probability distribution of points x of each class in the representation space $\mathbb{R}^d$. Hence, the representation of a class is given by some probability distribution in a vector space. The goal of the recognition (or *classification*) stage is to determine to which of its distributions the new vector should belong.

Adopting statistical notation, let random vector X denote the feature vectors which are "observables" of the process. The outcome of X is a concrete object representation x, mentioned above. Similarly, let Y be a random variable engendering some class representation y. The goal of the training step is to determine the class-conditional probability $P(X = x|Y = y)$, which, geometrically, is a probability density for points from class y being at position x. Conversely, the classification step is guided by the inverse conditional probability $P(Y = y|X = x)$ which can be interpreted as probability of the position x in the vector representation space $\mathbb{R}^d$ belonging to class y. In what follows, we briefly discuss the training and classification stages of the statistical pattern recognition framework. We show that class representation in this framework has two related interpretations: probabilistic and geometrical. In the first case, the class representation can be seen as a statistical distribution of points in $\mathbb{R}^d$ representing the class. In the second case, the class representation is a decision surface in $\mathbb{R}^d$ separating the region corresponding to the points representing a given class from the points representing all other classes.

1.1.3.1 Deriving Class Representation in a Vector Space

The usual dichotomy in pattern recognition consists of *supervised* and *unsupervised* modes of learning. Here we discuss the supervised mode, where the class membership of object in the training set is known. Unsupervised learning, such as clustering, is outside the scope of this discussion (the interested reader is referred to an overview by Jain *et al.*, 1999).

Let n be the number of classes in the environment. In other words, the outcome y of a random variable Y is a member of a set of n elements. Assume that y is fixed. To discuss the probability distribution of the points representing y, one must assume the existence of an infinite number of such points. In reality, to each class y there corresponds only a finite number of object representations from a training set. To simplify the discussion, we assume that the class-conditional probability only depends on the object representations belonging to y. This assumption allows the estimation of class-conditional distributions separately for each class. Hence, for the training stage, we fix X to denote the random vector representing a finite number m of training samples for y. Since y is fixed, we will refer to the class-conditional probability as simply $P(X|y)$. In general, there are two approaches to estimation of the class-conditional distribution — *parametric* and *nonparametric* (Devroye *et al.*, 1996; Watanabe, 1985).

In parametric approach, one usually assumes that the functional form f of the distribution $P(X|y)$ is known and the only unknown element in the specification of distribution is the parameter set θ. In other words, the probability of obtaining m feature vectors X assuming the probability density $P(X|\theta)$, given by configuration $f(X;\theta)$, is

$$P(X|y) = P(X|\theta) = f(X;\theta)\,.$$

Hence, the goal in parametric estimation is the search for parameters θ given the data X. The latter goal can be stated using the Bayes theorem as

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}\,,$$

where $P(\theta)$ is a *prior* probability of the parameter set and $P(X)$ is simply a normalisation term. The optimal parameter set $\widehat{\theta}$ is then given by the following maximisation

$$\widehat{\theta} = \arg\max_{\theta} P(\theta|X) = \arg\max_{\theta} P(X|\theta)P(\theta)\,. \tag{1.1}$$

The last term in the equation above is known as the *Bayes estimate* (sometimes also called *maximum a posteriori estimate*), which is logically the best estimate of the parameters one can hope to obtain (Devroye *et al.*, 1996; Kulkarni and Lugosi, 1998). An important property of the Bayes estimate is that it makes it absolutely explicit that in

order to obtain an analytically best estimate of the parametric configuration, one needs an additional *a priori* knowledge expressed by $P(\theta)$. Moreover, this prior knowledge is logically independent of the empirical evidence, expressed by the training data X. Thus, the sole assumption of the functional form f of the distribution is not enough. In practice, one can only make assumptions about $P(\theta)$ and hope that the estimation process using the resulting approximation will attain a set of parameters "close" enough to the ideal Bayes estimate.

Comprehensive review of various parametric estimation strategies is outside the scope of this thesis. In the remaining discussion we will only mention what is perhaps (historically) one of the most widely used estimates — the *maximum likelihood* estimate (Aldrich, 1997). One of the possible ways of obtaining a maximum likelihood estimate is to assume a uniform prior on the distribution $P(\theta)$. In other words, one fixes the latter term to some constant so that it becomes independent of θ and can be dropped from the estimation process in equation (1.1). This leads to the following optimal (in terms of maximum likelihood) parameter set

$$\widehat{\theta}_{\mathrm{ML}} = \arg\max_{\theta} P(X|\theta)\,. \tag{1.2}$$

Having fixed a functional form of $P(X|\theta)$, estimation of the parameters now becomes possible, as demonstrated by the following example:

Example 1.1 (Maximum Likelihood Estimation)**.** One of the most popular distributions for representing continuous data is the Gaussian multivariate distribution. In what follows, we give a very brief and informal example of estimating this distribution using a maximum likelihood approach.

Assume that the occurrences of the feature vectors in the training set, engendered by the random vector X, are independent and identically distributed. In addition, assume that the vectors are drawn from the multivariate Gaussian distribution given by (Deller *et al.*, 1993)

$$P(X|\theta) = f(X\,|\,\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^m|\Sigma|}}\exp\Big(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\Big)\,. \tag{1.3}$$

Without going into much detail (for details on this method see Aldrich, 1997), the optimal set of parameters $\widehat{\theta}$, corresponding to the maximisation of equation (1.3), is obtained as follows. First, a *log-likelihood* function

$$L(\mu,\Sigma) = \log f\big(X;\,\mu,\Sigma\big)$$

is defined. Then, one takes partial derivatives of L with respect to μ and Σ and equates them to zero. Solving the latter system of two equations yields the optimal (in terms

of maximum likelihood principle) parameter configuration, given by

$$\widehat{\theta} = \left(\widehat{\mu}, \widehat{\Sigma}\right) = \left((1/N)\sum_{i=1}^{m} x_i\,,\ (1/N)\sum_{i=1}^{m}(x_i - \mu)(x_i - \mu)^T\right),$$

where the $\widehat{\mu}$ corresponds to the mean feature vector of the training set and $\widehat{\Sigma}$ to its covariance matrix. ▷

Finally, we very briefly mention an alternative area of activity within statistical pattern recognition — *nonparametric estimation*. Unlike parametric estimation techniques, nonparametric estimation techniques make no assumptions about the type of class-conditional distribution $P(X|y)$. The nonparametric estimation is essentially concerned with construction of the probability distribution which fits the training data. This has been a vast and fascinating area of statistical pattern recognition since the earliest days. Perhaps the first and the most popular nonparametric estimators were the *histograms* and *decision trees*. The interested reader is referred to work by Kulkarni and Lugosi (1998) for an overview of nonparametric methods.

1.1.3.2 Classification in Vector Space

Assume the modelling environment consists of n classes. In other words, the class random variable Y has n outcomes, denoted y_i, $1 \leq i \leq n$. One is also given a set D of k previously labelled observations, represented by feature vectors x_j, $1 \leq j \leq k$. In what follows, the counter i ranges over a set of classes $\{y_i\}$, whereas the counter j over the training set objects $\{x_j\}$. The goal of the classification process is the construction of a *decision function* (or *classifier*)

$$\mathcal{D}_D(x) = y_l\,. \tag{1.4}$$

The above equation indicates that a new object representation x should be assigned to a class y_l on the basis of a training set D, where the optimal selection is the class with a maximum *a posteriori* probability

$$P(y_l|x) = \max_i P(y_i|x)\,. \tag{1.5}$$

The above equation can be specified in slightly more detail. Given class-conditional probability $P(x|y_i)$ one can obtain the above *a posteriori* probability

$$P(y_i|x) = \frac{P(x|y_i)P(y_i)}{P(x)}$$

using Bayes theorem. The only additional information which is required is the normalisation factor $P(x)$, such that $\sum_i P(y_i|x) = 1$, and the prior probability of the class $P(y_i)$. Substituting the above equation in (1.5) we obtain

$$P(y_l|x) = \max_i P(x|y_i)P(y_i)\,. \tag{1.6}$$

The above equation, known as *Bayes decision rule*, similar to the Bayesian estimate from the previous section, is the best decision that can be taken *given* that class-conditional and *a priori* distributions are completely specified.

In reality, one encounters problems with the accurate specification of the prior class probabilities and the nature of the statistical distributions of the training data D. The most likely cause for this is the sparsity of the training sample and insufficient prior information about the classes at hand. Hence, one hopes to construct classifiers which somehow mimic the optimal behaviour of the probabilistic Bayesian decision rule.

In the previous section, a statistical estimation approach was adopted for estimating the per-class conditional probabilities $P(x|y_i)$. If class-conditional density estimate *and* the estimate about the prior are given, the estimate of $P(y_l|x)$ can be obtained using the equation (1.6). This is demonstrated by the following simplified example:

Example 1.2 (Maximum Likelihood Gaussian Classifier)**.** Assume a uniform prior on all the classes. In other words, for each of the m classes y_i in the environment, $P(y_i) = \frac{1}{m}$. Given this assumption, the term representing the prior vanishes from maximisation in equation (1.6). Also assume that the functional form of the distribution $P(x|y_i)$ is a multivariate normal density. In this case, the maximum likelihood estimate of $P(x|y_i)$ is given by

$$P(x|y_i) = P(x|\widehat{\theta}_i) = \mathcal{N}(x;\, \widehat{\mu}_i, \widehat{\Sigma}_i)\,,$$

where the per-class optimal parameter set $\widehat{\theta}_i$ was derived in Example 1.1. Substituting the above equation in equation (1.6) we obtain the following classifier:

$$P(y_l|x) = \max_i \mathcal{N}(x;\, \widehat{\mu}_i, \widehat{\Sigma}_i)\,,$$

where the chosen class y_l corresponds to the parameter set $\widehat{\theta}_l = (\widehat{\mu}_l, \widehat{\Sigma}_l)$. ▷

When discussing nonparametric estimation methods in the last section, we mentioned the methods which attempt to construct a probability distribution whose type *and* parameters are induced by the training data (e.g. histograms). Overall, these approaches fall, to a certain extent, within the Bayesian framework because they involve estimation of probability distributions and their consequent use in the classifiers.

The parametric and nonparametric approaches, which attempt to model the densities directly, are often criticised for making quite unrealistic assumptions about the distributions (Watanabe, 1985, Section 9.1). In particular, it has been argued that for several tasks where the training sets are quite limited, the distributions are, in fact, not an appropriate way of modelling, since they assume existence of an infinite training sample for each of the classes. An alternative to density-based modelling is what is sometimes called a *geometric*, or curve fitting, approach (Jain *et al.*, 1999). In the

geometric approach, *decision surfaces* that *separate* the class distributions of points in $\mathbb{R}^d$ are constructed directly from the training data. These decision surfaces characterise the decision function $\mathcal{D}_D(x)$ from equation (1.4) on p. 9. The parameters of the decision surfaces are estimated during the supervised learning stage by optimising a certain error criterion over a training set. There are various optimisation criteria available (such as mean squared error between the current classifier output and the training target value), usually adopted from the area of nonlinear functional optimisation. Perhaps the most widely known are the two-class classifiers, such as Fisher's discriminant, single-layer perceptron, support vector classifiers, and others (Jain *et al.*, 1999; Kulkarni and Lugosi, 1998).

We conclude the discussion with one important observation. Statistical approaches employing the construction of decision surfaces are related to the density-estimation approaches for the following reason: a good fit for a decision function for some class y amounts to geometrically "encoding" the region of the maximum *a posteriori* class-conditional probability $P(x|y)P(y)$. In particular, it has been shown that Bayes classifiers (equation (1.6)) can in theory be approximated arbitrarily well by multi-layered perceptrons (cf. Kulkarni and Lugosi, 1998, Section VI).

1.1.4 Structural Class Representation and Generalisation

Structural approaches to generalisation can be divided into two groups: *syntactic* and *metric* (or *topological*) approaches.

1.1.4.1 Syntactic Approach

In syntactic approaches (Fu, 1982), one assumes the existence of a finite set of basic structures, called *atoms*, which can combine together using the *composition rules* specifying *a priori* domain-specific interrelationships between these atoms. The choice of atoms is important because on the one hand, they must possess the structure that can realistically be derived from the data and on the other, be rich enough to allow for encoding of complex objects and events using the composition rules.

When specifying syntactic representation and generalisation procedures, one usually draws an analogy between the structure to be represented and the theory of formal languages. In syntactic pattern recognition one assumes that any collection of observed object representations for a given class constitutes a realisation, or *language*, generated from some finite class description. One of the forms of class description is called *grammar*. A grammar is a rewriting system consisting of a finite set of atoms together with a finite set of composition rules. The syntactic approach is very appealing because of the following *generative* property: using a set of composition rules one can, in theory,

generate representations of an infinite number of objects belonging to a class.

In the early days of structural pattern recognition, it was believed that the notion of *grammaticality* holds the key to understanding the mechanisms of structural generalisation (see Tanaka, 1995). The notion of grammaticality originally appeared in linguistics due to Chomsky (1957) who introduced syntactic grammars as a compact mechanism for describing and generating grammatically correct sentences of natural language. This result has spawned the syntactic field of pattern recognition, where the notion of grammaticality can informally be stated as follows: given a grammar G_i describing a certain class, an object is said to belong to this class *if* it can be generated (or parsed) by that grammar. Objects belonging to other classes will not be parsed by G_i because they are "ungrammatical", i.e. outside the scope of a class description specified by a grammar. The task of generalisation within a syntactic pattern recognition approach is therefore an inference of a compact class description for a language which represents the training set. This task is often referred to as *grammatical inference*, where the goal if to infer a grammar. Alternative means of descriptions are possible if there *a priori* assumptions about the type of the language are made. For instance, if one is dealing with strings as object representation and the language L is assumed to be regular, the goal of generalisation is to infer a *finite state automaton* that represents L during the learning stage and then use this automaton to decide whether unknown strings belong to L during the classification stage. ,

The in-depth exploration of syntactic techniques is beyond the scope of this thesis. The interested reader may want to consult several chapters dealing with the syntactic approach in Bunke and Sanfeliu (1990). The study of grammars and automata for various classes of languages forms a vast field within the field of symbolic computation. Depending on the structure of the object representation, there are various theoretical results and techniques at one's disposal. Historically, the most studied structures are strings. For this type of structure, a vast array of formal results and tools, dealing with the representation and generalisation of the string languages, emerged over the years (e.g. Denecke and Wismath, 2002; Parkes, 2002; Sudkamp, 1997). The formal results (and as a consequence, applications) for other types of more complex structures (trees and graphs) are in a less mature state. For the current state-of-the-art in tree languages and automata (for which an excellent textbook by Gécseg and Steinby, 1984 is available), see work by Droste *et al.* (2005); Engelfriet *et al.* (2002); Fülöp and Vögler (1998, 2004). Similarly, the theory of graph grammars, languages and automata is still very much in development (some of the important results can be found in Ehrig *et al.*, 1999 and overview of recent state-of-the-art in Brandenburg and Skodinis, 2005).

To close this section, above we showed that the result of syntactic generalisation is

class representation which is essentially a grammar (or equivalent form of description, like finite state automaton). The type and formal properties of this class description are heavily dependent on the underlying symbolic modelling space, which is induced by the type of the object representation (strings, trees, graphs, etc.).

1.1.4.2 Topological Approach

In the general discussion concerning the concept of class it was mentioned that a class can be viewed as a collection (set) of object representations which are somehow related to each other. In topological approaches, this relation is more often than not expressed via the numeric measure of *similarity*. The latter notion appears so fundamental in pattern recognition that many researchers consider it to be the only scientific reality. The rather philosophically involved, and general in nature, arguments for primacy of a notion of similarity over that of classes and objects, definitely lie outside the scope of this discussion. The interested reader may want to consult work by proponents of this view, e.g. Edelman (1998; 1999). There is no doubt as to the fundamental role of the notion of similarity and nowhere is this role more evident than in the *topological* (i.e. similarity-based) approach to structural pattern recognition. The two mutually related and inseparable cornerstones of the latter approach are the object representation and the representation-specific *similarity measure*.

Let O be the set of object representations. One can define a special function f_O (whose properties are not discussed at this point), which given any two objects in O generates some numeric indication of how *morphologically* similar these objects are. In the topological approach, one usually assumes that the modelling environment is given by a pair M, informally called a *modelling space*, consisting of the above set of objects O together with a similarity measure f_O. The notion of a modelling space is conceptually related to the notion of a metric space which is absolutely pervasive in mathematics.

Similar to the syntactic approach, one assumes the existence (in the representation) of a set of basic and yet distinctive morphological structures (atoms) which describe the structural inter-dependencies in the representation of different objects. The main difference between the two approaches, however, lies in the requirement that the choice of structural inter-dependencies in the data O *induce* the similarity measure f_O. In other words, instead of treating the objects O in the representation as productions of some grammar, as it is done in purely syntactic approaches, in topological approach the objects are treated as "points" in some abstract symbolic space the properties of which are described by the similarity measure. Given the latter observation, one can hope to treat the problem of generalisation in a symbolic space similarly to the vector-space scenarios. This expectation turned out to be naïve, as we shall see below.

It should not come as a surprise that generalisation in abstract symbolic spaces is a very non-trivial task. The difficulties arise from the inherent structural complexity of the representation. In theory, one can hope to derive the "symbolic" analogues of the well-established analytical notions available in vector spaces. For instance, given some symbolic space (O, f_O), one can attempt to define the mean of set O, which is an object, not belonging to O but of a similar structure, with the same distance (or similarity) to all elements in O. This is a complex task, obviously dependent on the complexity of representation O, because it involves construction of a new object (e.g. string, graph, tree). In order to appreciate the dimensions of the difficulty (both analytical and computational) the reader is referred to some works dealing with the generalised notion of a mean for the case of strings (Nicolas and Rivals, 2003) and graphs (Jiang *et al.*, 2000), which are NP-hard problems for which various approximations were proposed recently.

Despite the obvious difficulties, the topological approach to structural pattern recognition is very popular, especially in those applications where modelling the structure of the domain is of paramount importance (e.g. bioinformatics Gusfield, 1997; Sankoff and Kruskal, 1983 and vision Edelman, 1999). The most fundamental question in these approaches is the choice of a similarity measure, which is heavily dependent on the structure of object representation. The structure of strings, being by far the most popular structural representation in pattern recognition, allows for several efficient ways of introducing the similarity, the most popular of which is the edit-distance proposed by Levenshtein (1966). Recently there has been a resurgence of interest in similarity measures defined on more complex structures, like graphs (Bunke and Jiang, 2000; Bunke and Shearer, 1998). The next fundamental issue is the issue of generalisation in the symbolic space M, mentioned above. In the topological approach, the class is usually represented as a small collection of prototype objects. During the classification stage, new objects are compared (by matching) to the prototype objects by using a predefined similarity measure. The classifier then assigns an unknown object to the class which contains the highest number of nearest (in terms of similarity) prototypes. The above classification rule, known as k Nearest Neighbours, is perhaps the most popular in structural pattern recognition. Its popularity stems from the fact that it is independent of object representation and, given a reasonable similarity measure, performs well in many different structural domains.

The above approach to representing classes as the prototypes is rather simplistic. Unfortunately, there is no general consensus in the pattern recognition community about the alternative ways of generalisation in the structural domain, which avoid the shortcomings of the syntactic approach (summarised by Tanaka, 1995), and yet utilise some of its most attractive features and realistic assumptions (e.g. see comments in Aiserman,

1969; Bunke *et al.*, 2001; Duin and Pękalska, 2005; Duin *et al.*, 2004; Goldfarb, 1990, 1992). We will return to the latter point in Section 1.3.

1.2 Representations in Speech Modelling and Recognition

At the beginning of this chapter, we mentioned that most of the advances in speech recognition research during the last two decades are often attributed to a flexible formulation of the speech recognition problem within a statistical pattern recognition setting (Jelinek, 1997; Young, 2001). In Section 1.2.1 we give a brief overview of statistical framework within which most of modern speech recognition research is conducted.

Similar to the analysis of the representations in a more general pattern recognition setting, described earlier in Section 1.1.2, in this section we introduce representations which are specifically used in speech recognition and modelling. As before, we consider two types of modelling spaces — numeric (Section 1.2.2) and structural (Section 1.2.3). The latter choice was motivated by the desire to compare and contrast the current speech recognition models with the models and formalisms we covered during the more general discussion in the previous section.

1.2.1 Speech Recognition Problem: An Overview

Speech recognition problem can be elegantly expressed within the statistical framework. Let $O = o_1^T$ be a sequence of acoustic observations. Without going into much detail about particulars of extracting these observations from the speech signal, at this point it suffices to mention that the above observations numerically encode in various ways the corresponding portion of the speech signal. The task of a obtaining observations from the speech signal is performed by a *speech signal processing front-end* of a recogniser, which, generally speaking, is not part of the statistical framework. In addition to the acoustic (physical) information described by O, let $W' = w_1^N$ be the sequence of words communicated by the speaker. The utterance W' corresponds to the acoustic record O. The goal of a speech recogniser is the find the most likely word sequence $\widehat{W}$ given the acoustic data O. If the recogniser does not make a mistake, $\widehat{W}$ will coincide with the word communicated sequence W'.

Within the probabilistic framework, which we briefly discussed in Section 1.1.3, we can formulate the latter statement as follows: Let $\{W\}$ be a finite set of allowable hypotheses produced by the recogniser. The probability of each hypothesis W given the data can be expressed as posterior probability $P(W|O)$. Within the Bayesian framework, the most likely hypothesis

$$\widehat{W} = \arg\max_W P(W|O) = \arg\max_W \frac{P(O|W)P(W)}{P(O)} = \arg\max_W P(O|W)P(W) \qquad (1.7)$$

is then chosen on the basis of maximum posterior probability. The above equation is sometimes referred to as the fundamental equation of speech recognition (Deng, 1998). The terms $P(O|W)$ and $P(W)$ in the last term of the above equation give rise to two major components in the statistical speech recognition framework.

1.2.1.1 Acoustic Modelling

The first major component of a speech recognition system is responsible for producing accurate estimates for $P(O|W)$, which is a probability of a certain word sequence *generating* a sequence of acoustic observations. These estimates are supplied by an *acoustic modelling* component. The acoustic model is an absolutely crucial component of any speech recogniser in that it tries to probabilistically bridge the gap between a sequence of linguistic signs, represented by W and the corresponding acoustic surface realisations O. This is usually accomplished by assuming the existence of an intermediate discrete-valued sequence of linguistic sub-word (or *sub-lexical*) models (or *units*) M, using which, the acoustic probability $P(O|W)$ can be factored as

$$P(O|W) = \sum_{M} P(O|M)P(M|W)\,, \tag{1.8}$$

where the probability is taken over all the sequences of sub-word models M. First, these models specify, according to probability $P(M|W)$, how words and words sequences can be expressed in terms of a particular linguistic organisation of a sub-word sequence M. Second, these models also provide information, in terms of probability $P(O|M)$, about how likely a particular sequence M is to produce surface acoustic observations O. The latter term $P(O|M)$ in the above equation (1.8) is often referred to as an *interface model*, where M provides the missing acoustic-linguistic link (e.g. see Deng *et al.*, 1997).

One of the simplest approaches to acoustic modelling is often referred to as *beads-on-a-string* (Ostendorf, 1999; Young, 2001). In this approach, a sequence M of K models corresponding to W is organised by linearly concatenating all the constituent sub-lexical acoustic models. The sub-lexical models can take many possible forms (phones, context-dependent phones, syllables and so on). The sequence (or more typically — sequences, in case a word has multiple pronunciations) of models for each particular word w is guided by the pronunciation lexicon.

The most widely used acoustic models are the Hidden Markov Models (HMMs). HMMs are essentially characterised by two random variables. The discrete sequence of K sub-word models needed to obtain W is modelled by an unobserved, and thus "hidden", discrete random variable S. This variable is called a *state variable*, whose outcome (state) s_t at each time instance t uniquely identifies the model m_i in a sequence.

The identification can be accomplished by assigning a label to each state, designating a model this state belongs to. Moreover, the state variable is assumed to possess the following property (called *Markov property of order 1*) (Bourlard and Bengio, 2002): All the information about the past of the system is summarised by the previous state only. In other words, the state *transition* probability can be written as

$$P(s_t|s_1^{t-1}) = P(s_t|s_{t-1})\,,$$

Let $S = s_1^T$ denote a particular state path through a sequence of M sub-word models. Using the above property $P(M, S|W)$ can be expressed as

$$P(M, S|W) = P(s_1) \prod_{t=1}^{T-1} P(s_{t+1}|s_t)\,. \tag{1.9}$$

An additional random variable, called the *observation variable*, corresponds to the observation sequence O and is dependent on the state variable S. However, because of the above Markov property of order 1, dependence of the acoustic observations on the state sequence S can be expressed as

$$P(o_t|s_1^t, o_1^{t-1}) = P(o_t|s_t)\,. \tag{1.10}$$

The above assumption is often referred to as the *conditional independence* property: the observation is only dependent on the current value of the discrete state variable. Hence, the probability of a model M generating a sequence of acoustic observations O by following the state sequence S can be estimated as

$$P(O, S|M) = \prod_{t=1}^{T} P(o_t|s_t)\,,$$

where $P(o_t|s_t)$ is called an *emission probability*.

Example 1.3 (Hidden Markov Model)**.** In speech recognition literature, by *structure* of an HMM-based model one usually understands a sequence of the values of the state variable S linked by nonzero transition probabilities. An overall structure is thus referred to as *topology* of a model.

Topology of a typical single phone (*monophone*) HMM is shown in Figure 1.1. A *phone* is a smallest perceptible discrete sound segment in a speech stream. The state sequence for this model is organised in left-to-right and self transition arrangement. The actual acoustic model consists of three emitting states s_2, s_3 and s_4. The auxiliary non-emitting states s_1 and s_5 are needed for connecting this model to other models. Transition probabilities $P(s_i|s_j)$ are denoted $a_{j,i}$ and the emission probabilities $P(o_j|s_i)$ by $b_i(o_j)$. Six observations are "emitted" by three states in this example.

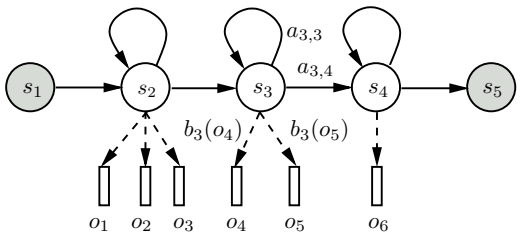


Figure 1.1: A typical single phone (*monophone*) HMM (after Young, 2001), where the state sequence is organised in left-to-right and self transition arrangement. The actual acoustic model consists of three emitting states s_2, s_3 and s_4. The auxiliary non-emitting states s_1 and s_5 are needed for connecting this model to other models.

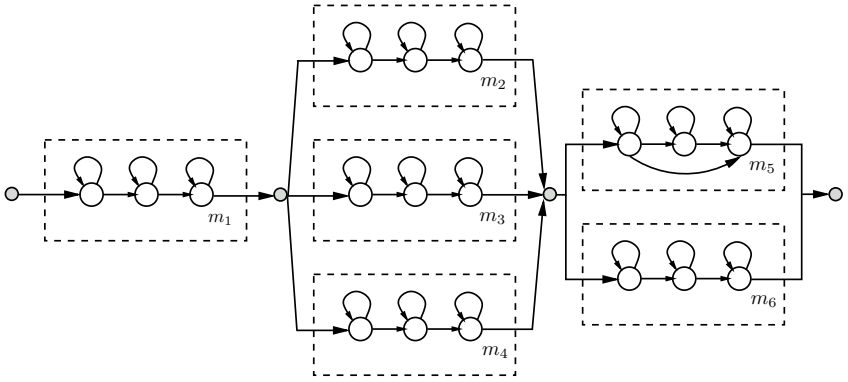


Figure 1.2: Simple word-level HMM model formed from constituent sub-word HMM models according to pronunciation dictionary.

The single phone (or other sub-lexical) models are concatenated together according to pronunciation dictionary to form a compound HMM model for some word w, as shown in Figure 1.2. According to the model shown in the figure, the structure of word w represents various (ordered) sequences of three models $\{m_1, m_i, m_j\}$, where $m_i \in \{m_2, m_3, m_4\}$ and $m_j \in \{m_5, m_6\}$. Note that the structure of the sub-word models is not necessarily fixed (model m_6 has a structure different from the rest of the models). ▷

In light of the above, within the sub-lexical HMM framework, the overall acoustic estimate $P(O|W)$ is not only dependent on all possible sequences M of the sub-word models, but also on all the possible state transition sequences S within these models. In other words, equation (1.8) can be rewritten as

$$\begin{aligned} P(O|W) &= \sum_M \sum_S P(O, S|M) P(M, S|W) \\ &= \sum_M \sum_S P(s_1) \prod_{t=1}^{T-1} P(s_{t+1}|s_t) \prod_{t=1}^{T} P(o_t|s_t) \,, \end{aligned}$$

where the sum is taken over all the pronunciation sequences M and over all the possible

(non-zero) state paths S in each of the pronunciation sequences.

There is a strong correlation between pronunciation variation for each word w in a sequence W and the type of the sub-lexical units (called *baseforms*) which represent the structure of w. Usually, the more sophisticated the baseform is, the bigger the number of possible paths through the word model. The latter point is demonstrated by the following discussion where we compare and contrast two different types of sub-lexical models. For HMMs representing single phones (Example 1.3), since there are around 45 distinct phonemes in English (Young, 2001), pronunciation variation is usually constrained by a small number of possibilities for each word. The more accurate sub-word models take into account the contextual influence of the previous and following number of phones on the realisation of a given phone. The most common type of such context-dependent models is a *triphone*, where one previous and one next phones are taken into account. Thus it is possible to better model the pronunciation variation, e.g. [s-t-oh] in "stop" versus [ae-t-sil] in "that", where the latter realisation of [t] sound is usually unaspirated at the end of the word in American English (Jelinek, 1997). Compared to monophones, the number of possible context-dependent units grows exponentially with the size of the context. For example, there are 45^3 triphone models corresponding to 45 single phone models. Obviously, large proportion of triphones is ruled out by phonological rules (e.g. clusters like [zh-z-zh] are clearly not encountered in English). There is still, however, a significant number of legal combinations that are either not observed in the data or appear rarely. In order to deal with this problem, often referred to as *data sparsity*, efficient techniques were developed to automatically reduce the number of model parameters by the use of various clustering techniques (perhaps the most popular is the *state-tying* strategy, overviewed by Young (2001)).

We will return to acoustic models in Section 1.2.1.3, where they will be considered in the context of generalisation.

1.2.1.2 Language Modelling

The prior probability $P(W)$ in the "fundamental" equation (1.7), specifies how probable a word sequence W is, based on some *a priori*, *supra-lexical*, information that is independent of acoustic observations. This estimate is produced by a *language modelling* component. Informally, the task of a language model is to indicate which hypothesised word sequences are likely to be encountered in the natural language based on some syntactic and semantic information. Consequently, the estimate produced by a language model for a given hypothesis W will weight down the overall estimate of $P(W|O)$ if the hypothesis W is not very likely to be grammatical. Alternatively, it will boost the confidence of the estimate if the hypothesis is likely to be legal. Initially, statistical

language modelling was considered to be solely within the domain of speech recognition research and statistics. For some early applications of language modelling to speech recognition see the works of Bahl and Jelinek (1989; 1990; 1985). The earliest, the simplest and also historically the most widely used supra-lexical model is the stochastic n-gram language model which specifies the probability of the next word given its $n-1$ contexts. The size of the context is intentionally limited (usually to be not larger than 3) in order to make estimation of the individual probabilities from the data tractable. Using an n-gram model, the probability of a word sequence $W = w_1^N$ can be expressed using as

$$P(W) = \prod_{i=1}^{N} P(w_i|w_1^{i-1}) = \prod_{i=1}^{N} P(w_i|w_{i-n+1}^{i-1})\,.$$

The above language model is perhaps the simplest of all possible because it relies solely on the word distributions and ignores the long-range word dependencies and any explicit *a priori* syntactic and semantic knowledge. Nevertheless, it proved to be surprisingly reliable and still forms the language modelling backbone of many state-of-the-art modern commercial and research speech recognition systems.

In HMM-based speech recognition framework, language models allow to link the word-level HMM models by transition probabilities which represent the language model estimates. The simplest structure may correspond to the case when the language model is memoryless, i.e. the probability of any given word is independent of the history. In this case one can simply combine all the models of words from some lexicon in a parallel loop. In practice, in order to obtain reasonable estimates, at least a trigram (n is 2) model should be used. This complicates the decoding process since the overall network has to maintain all the possible word histories of length $n-1$. In general, the longer the list of word dependencies of a language model, the more complicated is the overall network (the number of states in the overall framework is proportional to V^{n-1}, where V is the size of vocabulary). Partial transition structure for recognition network employing trigram language model is shown in Figure 1.3.

During the last decade, due to the success of the statistical language modelling techniques they also formed a vast research area within the field of computational linguistics (see an overview by Manning and Schütze, 1999). Language modelling has also influenced the development of statistical approaches in other areas of pattern recognition, such as handwriting recognition (Lui *et al.*, 2003). Although language models *per se* are outside the scope of this thesis, we will nevertheless return to them later on in this section and consider them in slightly more detail in the context of decision making within probabilistic speech recognition framework.

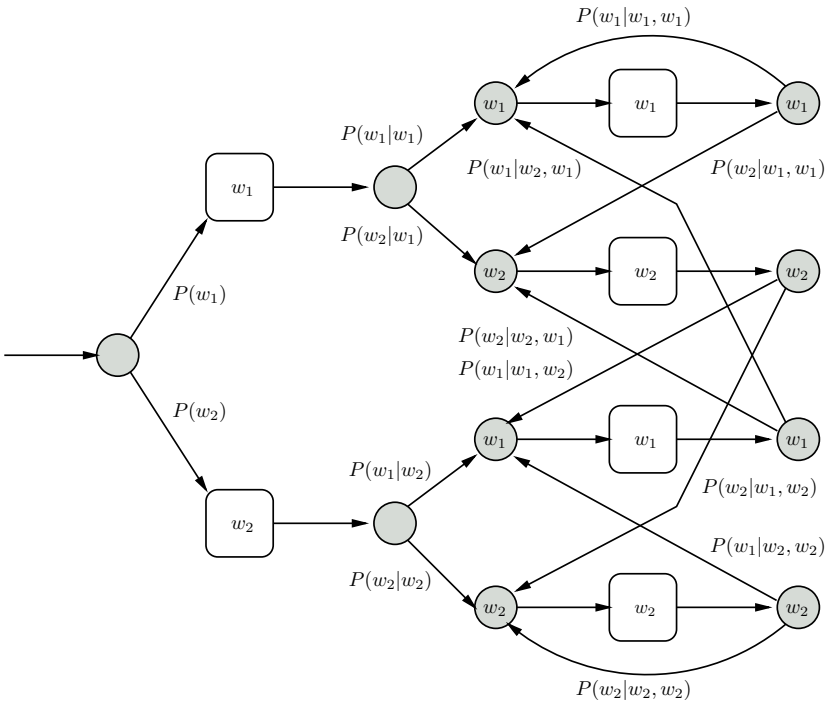


Figure 1.3: Partial transition structure of an HMM-based speech recognition network based on a trigram language model. The vocabulary consists of two words w_1 and w_2. The boxes represent HMM-based word models, like those shown in Figure 1.2 (after Jelinek, 1997, Figure 5.3).

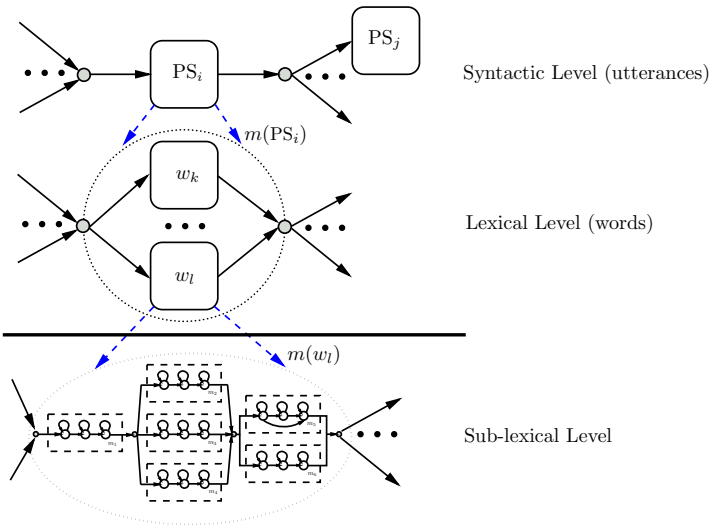


Figure 1.4: A simplified illustration of hierarchical organisation of speech knowledge within a hypothetical HMM-based recognition network. The dashed arrows indicate internal organisation of a unit in question. The thick line divides the organisation of sub- and supra-lexical levels. The syntactic layer units (PS$_i$, etc.) indicate parts of speech and are shown for illustrative purposes only.

1.2.1.3 Categorisation in Statistical Speech Recognition

We open this section by briefly discussing the recognition stage of the generalisation process. One of the most attractive features of speech recognition models (such as HMMs) that are based on the discrete state random variable is the fact that these models can be seamlessly combined together to obtain complex compound state-space structures, which are amenable to the same analysis as the constituent models. In the previous section we already mentioned the integration of various sub-lexical (acoustic) and supra-lexical (language) knowledge sources into one compound HMM model (Figure 1.3). In general, in order to represent various knowledge sources associated with a particular speech recognition domain, one usually compiles a large *recognition network*, prior to recognition stage. Schematic representation of hierarchical knowledge within an overall HMM-based recognition network is shown in Figure 1.4.

During the discussion of HMMs, we mentioned that to each state in a model there corresponds a classification label which uniquely identifies a unit (e.g. phone) it belongs to. Hence, in an HMM-based recognition framework, to each state sequence corresponds a sequence of classification labels. The latter fact allows the clarification of objectives for the recognition stage (often referred to as *decoding*): Given an acoustic stream $O = o_1^T$, obtaining an optimal word hypothesis $\widehat{W} = \widehat{w}_1^N$ amounts to finding a state sequence $\widehat{S} = \widehat{s}_1^T$ maximising posterior probability given the observations. The

search is conducted in a recognition network (mentioned above) that represents all the relevant linguistic knowledge in one compound HMM. In general, the decoding problem is hard because of the inherent complexity of hypothesis space and the "straightforward" dynamic programming approaches are often computationally intractable. The standard state-of-the-art schemes use multiple passes over the recognition network. At each pass, rather than producing the most optimal sequence, the decoder outputs a *lattice* of word sequence hypotheses. A lattice essentially indicates which words were likely to be uttered at which time intervals. Thus one can constrain the search space for the next pass. Interested reader is referred to reviews of decoding state-of-the-art in (Aubert, 2000; Jelinek, 1997; Young, 2001).

The computational (in terms space and time) cost of the recognition step can be greatly reduced by using an alternative approach employing *finite state transducers*. Finite state transducers can efficiently and unambiguously represent the recognition network in an elegant mathematical framework without the loss of original information. In his paper, Mohri (1997) mentioned that for a 10-word sentence from an ARPA ATIS task, the recognition network consisted of nearly 83 million paths. When encoded as a finite state transducer and optimised, the resulting lattice consisted of 18 paths. Thus, conducting the search on the optimised lattice greatly improved the performance. For more information on finite state technologies in speech recognition the reader is referred to recent reviews and applications in (Hazen *et al.*, 2005; Mohri *et al.*, 2002). We will briefly return to the discussion of finite state devices later on in Section 1.2.3.2, where we consider them in a structural pattern recognition setting.

We next consider another stage of generalisation in speech recognition that deals with learning. Nearly all of the approaches to statistical speech recognition employ parametric estimation techniques. These techniques were considered in a more general pattern recognition setting in Section 1.1.3. In speech recognition, the most commonly encountered learning target is the maximisation of $P(\mathcal{O}|\mathcal{W},\theta)$, where

- The training data $\mathcal{O}$ consists of a set of acoustic observations $\{O_i\}$ corresponding to a set $\mathcal{W}$ of utterances $\{W_i\}$;
- The model parameter set is given by θ.

The traditional approach to training is to consider each model (e.g. word-level model) independently and to estimate the distribution parameters of *subsequences* of acoustic observations which correspond to that model. In HMMs, for instance, when training with a particular word, the path through the compound HMM representing the training set utterance is *constrained* so that it can pass only through that word's model. After the several complete passes through the training data, the parameters of the overall

network representing the training set are updated with an optimal parameter set $\widehat{\theta}$.

As mentioned in Section 1.2.1.1, the parameter set of the HMMs consists of the transition probabilities and emission distributions. Since the state transitions are the outcomes of a single discrete random variable S, they can be simply represented by a transition probability matrix $A_{i,j} = P(s_{t+1} = j|s_t = i)$. Emission probabilities are usually modelled by a *Gaussian mixture*

$$P(o_t|s_t = i) = \sum_k c_{i,k}\mathcal{N}(o_t; \mu_k; \Sigma_k)\,,$$

where $c_{i,k}$ is the k-th mixture non-negative weight (constrained so that the weights of all mixtures sum to unity), and $\mathcal{N}(o_t\ \mu_k; \Sigma_k)$ is a k-th Gaussian distribution (parametrised by a mean vector μ_k and the covariance matrix Σ_k) specifying a probability of observing vector o_t. Modelling the observation vector with the mixture of Gaussians attempts to model correlation structure in the observation vector o_t. Thus, the parameter set θ for HMMs, can be represented as a three-tuple consisting of transition matrix, a set of mean vectors and a set of covariance matrices.

In statistical speech recognition, there are various parametric parametric approaches to learning which usually mirror state-of-the-art in machine learning and pattern recognition. Traditional parametric learning approaches employ maximum likelihood criterion for estimation of the parameter set θ (this criterion was briefly described in Section 1.1.3). In maximum likelihood approach to speech recognition, one chooses the parameter set θ^x which maximises the product of the class-conditional likelihoods $P(\mathcal{O}^x|\mathcal{W}^x, \theta^x)$ over the training sequences representing some class x (this could be almost any non-trivial speech unit). In other words,

$$\widehat{\theta}^x = \arg\max_{\theta^x} L(\theta^x) = \arg\max_{\theta^x} P(\mathcal{O}^x|\mathcal{W}^x, \theta^x) = \arg\max_{\theta^x} \prod_i P(O_i^x|W_i^x, \theta^x)\,.$$

The above optimisation can be efficiently implemented by several standard algorithms, the most popular of which is known as *expectation-maximisation* (EM) algorithm. The details of this algorithm can be found in any standard textbook on speech recognition, such as (Jelinek, 1997) and (Deller *et al.*, 1993).

The above parametric approach is often called *non-discriminant* because it deals with an estimation of the acoustic distributions corresponding to a particular speech unit (e.g. word) rather than attempting to estimate the acoustic differences between various classes of speech units. This observation led to the emergence of various *discriminant criteria* for the training phase of speech recognisers. These criteria include the maximum *a posteriori* (MAP) optimisation of $P(\mathcal{O}|\mathcal{W}, \theta)$ and the estimation based on the maximum mutual information (MMI) optimisation of $\log \frac{P(\mathcal{O}^x|\mathcal{W}^x)}{P(\mathcal{O})}$. Optimising the MMI criterion essentially amounts to simultaneously increasing the likelihood of

the constrained model $P(\mathcal{O}^x|\mathcal{W}^x)$ while decreasing the likelihood of the unconstrained model that represents all the word sequences. The estimation algorithms for implementing the above criteria are generally more complicated than the ones used for maximum likelihood estimation. These algorithms use gradient descent approaches adopted from non-linear functional optimisation (see Bourlard and Bengio, 2002 for an overview).

1.2.2 Numeric Representations

In the previous section we reviewed some of the fundamental components of mainstream speech recognition systems. As mentioned earlier, the factor that contributed most to the advances in performance of such systems is a statistically sound formulation of speech recognition problem. In this section we briefly outline some current and existing research aimed at improving the modelling and recognition process. Precise classification of the following approaches and research directions is difficult because there is a significant overlap and mutual influence between them. The need for alternative approaches is clearly manifest in the considerable amount of literature describing new models (e.g. see an editorial by Russell and Bilmes, 2003 and upcoming special issue on non-conventional techniques in Faundez-Zanuy *et al.*, 2004).

The majority of mainstream speech recognition systems are based on the HMMs, which we briefly described in the previous section. The key feature of HMMs that was criticised by many researchers (e.g. Young, 2001) is the *frame independence assumption*, whereby each successive speech feature vector is assumed to be conditionally independent given the HMM state (equation (1.10) on p. 17). Deng *et al.* (1997) argued that such assumptions are too severe and discard many of the key temporal correlation properties in the speech signal, which result from relatively smooth movement of articulatory structures during the act of speech production. Moreover, it was observed that these correlation properties are not frame-specific but rather depend on segments. A segment is some variable-length linguistic unit, usually a phone. While the acoustic correlation within the same segment is usually high, cross-segmental correlations are lower (Glass, 2003).

A desire to weaken the above assumption motivated the development of *segment models* (Ostendorf *et al.*, 1996). In segment-based modelling, it is convenient to think of a model as generating segments o_1^l of random length l, rather than individual feature vectors, which are still used for representing the speech signal. In probabilistic terms, a segment of random length l representing some phone m is given by

$$P(o_1^l, l|m) = P(o_1^l|l, m)P(l|m)\,,$$

where $P(o_1^l|l, m)$ is the observation distribution provided by acoustic model and $P(l|m)$

is the estimate provided by the duration model. Several approaches have been proposed for modelling the observation distribution characterising a segment. Some use functional parametrisation using trajectory functions (Holmes and Russell, 1998; Young, 2001). The other approaches describe the above observations via linear dynamical systems where the trajectory over the segment is described by a continuous-state variable from which the observations are derived (see an overview by Frankel, 2003). In yet another approach, used in the SUMMIT speech recognition system, the variable-length segments are not based on speech frames but rather on *acoustic landmarks*, which are various asynchronous acoustic events detected in the signal (see the work of Glass, 2003 and his group).

Another frequently criticised property of the HMM-based systems is the *beads-on-a-string* assumption (Ostendorf, 1999), described in Section 1.2.1. It was argued that, as far as everyday spontaneous speech is concerned, this configuration fails to model significant phonological variations which are otherwise easily accounted for by modern phonological theories. In phonology, the phonological processes responsible for this variation are modelled by parallel *feature streams* (or *tiers*). A certain variation in the realisation of some phoneme is not modelled by the substitution of one segment (phone) for another, as it is done in beads-on-a-string approach, but rather by difference in timing of asynchronous feature changes in some of the feature streams. The adjective "asynchronous" above corresponds to the observation that features rarely change at the phone boundaries. Hence, the streams cannot be "lined up" (Livescu *et al.*, 2003) to form phonetic segments (phones). The above observations motivated various studies which, on the one hand, focus on modelling "hidden" multiple parallel asynchronous processes (e.g. factorial HMMs used by Nock, 2001, Dynamic Bayesian Networks used by Livescu *et al.*, 2003, nonlinear dynamic models used by Deng, 2000) and on the other focus on the automatic mapping between the acoustics and phonological feature space in which acoustic modelling can be conducted (e.g. work by King and Taylor, 2000; King *et al.*, 2000). Another area of research is the incorporation of a full nonlinear dynamic speech production model motivated by the theory of articulatory phonology (this work was primarily conducted by Deng, 1998 and his group).

Another open research direction is the issue of learning the appropriate topologies of the acoustic models which may better suit the speech data. In traditional approaches, topology (or structure) is usually understood as representation of transitions of a hidden state variable. Such representation is crucial because it captures temporal dependencies of the process being modelled. In most of the conventional approaches a significant amount of *a priori* knowledge influences the choice of model structure before considering the actual learning process. Some promising results in the area of statistical inference

of acoustic model structure were recently reported by Zweig *et al.* (2002) in the context of graphical models. Graphical models are powerful statistical graph-based abstractions flexible enough to encompass nearly all the statistical models proposed to date (see an excellent overview in Bilmes, 2003).

1.2.3 Structural Representations

In this section we briefly review some common structural models which are used in speech modelling and recognition.

1.2.3.1 Annotation Graphs

Various graph-based paradigms used for annotating speech corpora are collectively referred to as *annotation graphs*. The emergence of general-purpose frameworks for annotating speech was motivated by the apparent lack of standards, the need for which was acutely felt within the speech community (Bird and Liberman, 2001). The proponents of the above standardisation hope that the adoption of a unifying *formal* approach to representing speech corpora will greatly facilitate research within the field.

Annotation graphs cover any descriptive or analytic notation applied to raw speech data. The notations might come from a wide spectrum of sources ranging from phonological features to discourse structures, morphological and syntactic analyses, word senses, semantic relations and so on. Several speech annotation frameworks have been suggested up to date, among which we would like to single out the most flexible ones: *annotation graphs* (AG) by Bird and Liberman (2001), and *heterogeneous relation graphs* (HRG) by Taylor *et al.* (2001). HRGs have seen extensive use in speech synthesis applications as representational devices (e.g. see Taylor and Black, 1999), while AGs so far have only seen use in annotation.

Among the multitude of available multilayer graph-based formalisms, we can discern the single common most important feature: the ability to associate a label or an ordered set of labels with a stretch of time in the recorded speech (Bird and Liberman, 2001). An additional important advantage offered by these formalisms is that each utterance is represented by a single multilayer graph structure, in which various knowledge sources are unified. This graph structure is obviously fully amenable to linguistic analysis because it is created either by a human expert or constructed automatically (as it is the case with the text-to-speech synthesis systems) from various previously annotated sources[3].

[3]The research work on this thesis initially started from an attempt to model HRG structures in a graph transduction framework, which, in theory, would have allowed efficient (time and space-wise) encoding of the various knowledge sources used in text-to-speech engines (the idea due to Paul Taylor).

Although annotation graphs are very attractive in terms of their capacity of representing the variety of linguistic information associated with the speech waveforms, these frameworks are unfortunately not suitable as pattern recognition models. This inadequacy is currently due to the following factor: from the formal point of view, since the graph structures used in annotation frameworks incorporate heterogeneous information, it is not clear how to treat them as object representations. In the discussion of structural pattern recognition in Section 1.1.4, we mentioned that in order to conduct any kind of structural generalisation using annotation graphs, one essentially needs to introduce either:

- a graph grammar for annotation graphs, which, in theory, will allow the application of syntactic pattern recognition techniques, or
- a similarity measure defined on a set of annotation graphs, which will allow the introduction of a notion of symbolic space and application of structural pattern recognition techniques (e.g. nearest neighbour analysis).

The first option is hard due to the high inherent complexity of graph representations (we briefly mentioned this issue in Section 1.1.4.1). In addition, since representation has multiple semantically distinct layers, it is not clear how to combine this information in a single grammar. The latter is also the reason for the impracticality of the second, similarity-based, option. It is not clear whether definition of a real-valued similarity mapping on annotation graphs is linguistically meaningful.

1.2.3.2 Finite State Transducers

In this section we briefly review the use of *finite state transducers* in speech recognition (which we already mentioned in the context of decoding in Section 1.2.1.3). In general, a finite state transducer (FST) is a finite state machine whose state transitions are labelled with a pair consisting of input and output symbols from some finite alphabets (Parkes, 2002; Sudkamp, 1997). Any path through a finite state state transducer algebraically encodes the mapping from a sequence of input symbols to a sequence of output symbols. An important modification of the FST architecture is obtained if the output of each transition is also allowed to have some numeric value associated with it. The resulting configuration is called a *weighted finite state transducer* (WFST). A WFST, in addition to encoding the symbolic transduction also allows computation of an overall cost of the corresponding path through the model. A WFST whose transitions are weighted but contain no output symbols, is called a *weighted finite state acceptor* (WFSA).

As mentioned earlier during the discussion of decoding techniques, in speech recognition, finite state machines (mostly WFSTs and WFSAs) have become very popular for

representing various knowledge sources (language models, pronunciation models, acoustic models and so on), in a single formally homogeneous recognition network, which in itself is a finite state machine. This popularity is due to the fact that the mathematical (algebraic) theory of the finite state machines is very powerful (at least for the case of string transduction) and, in particular, provides efficient means for drastically reducing the size of the original FST representing the recognition network (this is called *minimisation*) and also removing structural ambiguities (resulting in exactly one transduction path per input sequence) in the original FST (*determinisation*) (Mohri, 1997; Mohri *et al.*, 2002). The performance of the various approximating search algorithms used by the decoders is often dramatically improved if, instead of an unoptimised network, the search is conducted in a recognition network optimised using the above techniques. This is caused by a large decrease in the complexity of a search space (during the FST optimisation, all the redundant transition paths are removed and many paths are shortened).

Recently, parametric optimisation techniques were extended to weighted finite state transducers. Among these is an expectation-minimisation (EM) algorithm for WFSTs (Eisner, 2002). Given the fixed WFSTs configuration and a training set of input/output symbolic sequences, EM algorithm infers a set of optimal transduction weights associated with the training sequences. Trainable WFSTs are becoming more popular in pronunciation modelling. One interesting application of WFSTs to pronunciation modelling is the incorporation of the likelihoods of alternative pronunciations of various words in a lexicon (traditional lexical FST-based models assume that all the pronunciations are equally probable). While some of the alternative pronunciations are more likely to be encountered in practice, the others are highly improbable. Incorporation of a likelihood model ensures that the recogniser picks the more probable hypothesis. Inference of likelihoods of alternative pronunciations by training the pronunciation WFSTs was recently demonstrated by researchers working on the SUMMIT speech recognition system (Hazen *et al.*, 2005; Shu and Hetherington, 2002).

To summarise this section, in speech recognition finite state transducers of various types (FSTs, WFSTs and so on) are used primarily as efficient *knowledge representation* devices which combine various knowledge sources. These knowledge sources, in turn, can be seen as finite state devices themselves, because they are often based on similar Markovian assumptions (HMMs in acoustic modelling, n-grams in language modelling are typical examples of such models). If we compare this aspect of transducers to the models used in structural pattern recognition, FSTs bear certain similarity to syntactic approaches because historically, finite state machines appeared as "automated" means of computing various formal languages (Sudkamp, 1997). Similar to grammatical inference

(Section 1.1.4.1), *structural* inference of automata and transducers from the data is a hard problem, the complexity of which grows with the complexity of the structural object representation. Within the statistical framework of speech recognition, since the representation space is a vector space, structural inference of transducers is simply not possible. Instead, the task of interfacing with the object representations (expressed as feature vectors) is delegated to acoustic models of inherently numeric nature. Hence, in the context of *speech* representation and generalisation, the role of transducers can be considered to be secondary.

1.3 Research Motivations

1.3.1 Current Situation with Representations

As was shown in Section 1.1, the concept of representation is absolutely crucial in pattern recognition. The representation of real world objects or events is supposed to encode them in some mathematical framework that has the capacity of relating these encodings to one another. Furthermore, as a consequence, generalisation is achieved by deriving some compact discriminating descriptions of classes represented by the related encodings. Despite the critical role the notion of representation is supposed to play, traditionally it has been frequently neglected. As was aptly observed by Duin and Pękalska (2005), many relatively recent books on pattern recognition and machine learning disregard the issue of representation altogether by assuming that the representation is somehow provided by some outside expert knowledge (e.g. Bishop, 1995; Ripley, 1996). However, there is a certain danger in placing the notion of representation outside the scope of the overall generalisation framework, because representation and generalisation are not independent. The choice of the representation essentially induces the modelling framework in which the generalisation can be approached. Moreover, the process of generalisation itself involves representation of the classes.

Earlier in this chapter, an overview of current approaches, both structural and numeric, to representation in pattern and speech recognition was provided. The ongoing debate in artificial intelligence on which one of the approaches is better suited for modelling human intelligence, is definitely outside the scope of this thesis. The philosophical arguments put forward by proponents of either approach are certainly within the domain of cognitive science, psychology and philosophy (for example, see Stender and Addis, 1990). These arguments are not very productive from the applied point of view because the value of the theory is usually in its utility. In contrast to the field of artificial intelligence, in pattern recognition it is generally agreed (see remarks by Kanal, 1993, Pavlidis, 2003 and Goldfarb and Nigam, 1994, for instance) that there is no single

best approach and any complex domain often requires a combination of both. In what follows, we briefly summarise our observations with regard to the two approaches to *representation* in pattern recognition (and speech recognition in particular) and attempt to outline some of the open issues. When talking about representations, we consider both the representations of objects and the representations of classes.

1.3.1.1 Open Issues in Numeric Modelling

In the previous sections of this chapter we mentioned that numeric representation is essentially an embedding of the data into some d-dimensional vector space, whereby each object is represented as a point in $\mathbb{R}^d$. Categorisation is then performed using mathematical decision-theoretic tools available in vector spaces, such as estimation of density functions underlying the clusters of points (Section 1.1.3). Once the representation is fixed, there is a multitude of analytical tools available in vector spaces that can be used for constructing models for generalisation. In the context of speech recognition, the representations correspond to sequences of acoustic feature vectors, while the generalisation mechanism is provided by the statistical learning and recognition framework (Section 1.2), where the interface between the representation and generalisation is provided by the acoustic models. In the context of study of representations, we briefly outline some of the open issues with the feature vector-centered approach. More often than not, some of the issues mentioned below are dictated by the nature of the modelling space $\mathbb{R}^d$ rather than by ill-formed assumptions:

First, the most often heard criticism of the feature vector representation is that it is too restrictive. Any spatial, temporal and other relations between the "parts" of the original object or event are usually not preserved by the reduction of an object to a vector. This criticism is especially relevant when it is generally agreed that the domain in question contains some well-identifiable structure (e.g. genomes in bioinformatics, characters in optical character recognition). Certain original relations are usually partly recovered statistically during the generalisation, however these relations are not present in the representation itself. As a result, the feature vectors are often not interpretable without the generalisation framework. Reduction of the original complexity of the domain to feature vectors, shifts the emphasis to introducing some of the lost structural relations into the generalisation framework. It is interesting to note that some simpler models in speech recognition, such as HMMs, are often criticised for not possessing enough structure (Deng, 1998, 2000), where the structure is often interpreted in statistical terms. In the words of (Jelinek, 1997, p. 10), such models "... have no more than a mathematical reality. No claims whatever can conceivably be made about their relation to human's actual speech production or recognition".

Second, the elements of a feature vector are often mutually incommensurable. This is problematic because the entire generalisation framework in vector space rests on the Euclidean assumption that all the dimensions in the representation have equal weight. In physics, for example, the problems with the above assumption led to emergence of special (Minkowski) vector spaces in which the incommensurable dimensions are "decorrelated" in order to accommodate for space-time vectors (Pyenson, 1977). One of the popular representations in speech recognition is based on the *mel-cepstrum*, which is well-motivated by studies of human auditory perception (Deller *et al.*, 1993). Without going into much detail about derivation of this representation, it suffices to note that more often than not, the vector representation of mel-cepstrum, called the mel-cepstral coefficients (MFCCs) often includes delta and delta-delta mel-cepstral information, because this information was observed in practice to improve the recognition performance. In terms of a Euclidean modelling space, however, this inclusion can hardly be justified on mathematical grounds.

Before formulating the third objection to feature vectors as the basis of representation, a following observation is in place: once fixed, the mathematical properties of object representation essentially dictate the ways in which the next generalisation stage (involving learning and recognition) can proceed. In other words, one of the results of the generalisation stage is the representation of the classes in question, which in turn is heavily influenced the original representation of the objects. The generalisation processes in vector spaces have been extensively studied over the years and a multitude of diverse techniques were developed (Jain *et al.*, 2000; Kulkarni and Lugosi, 1998). What is a class representation in vector space? As was shown in the previous sections, class representations in vector spaces are essentially defined by the parametric distributions describing clusters of points, decision surfaces which separate these clusters or surfaces produced by parametric curve fitting. To what degree such a representation of classes is informative or adequate (in terms of interpretation) is a matter of dispute. The author believes that in linguistic and artificial intelligence terms such class representations are suboptimal. Some highly nonlinear separating hypersurface in d-dimensional space does not tell one much about the morphological makeup of the original classes in question and it is not difficult to find a linguist who might object to such a notion of a class (in other areas, similar concerns were voiced by Abela, 2001; Davis and Shrobe, 1993; Goldfarb, 2004, and others).

1.3.1.2 Open Issues in Structural Modelling

Structural approaches to representation and generalisation were described in Section 1.1.4 (for general pattern recognition) and in Section 1.2.3 (for speech recognition). As was

mentioned in the discussion of structural approaches, their main benefit is that (well-formed) structural encoding of objects retains the original relations between various constituent parts of an object. This is because the structural modelling space induced by the representation is structurally more expressive than a regular vector space. Moreover, in theory, in a structural modelling space the introduction of a non-trivial class representation should be possible. The above benefits, however, come at the following costs:

It is not clear on which foundation such structural representation should be based. Unlike vector space approaches, there is no fixed modelling space induced by the chosen representation. This issue is definitely not trivial. On the one hand, the choice of a more complex representation would allow the encoding of more complex relations within an object. On the other hand, since the representation is based on real data, the more complex the representation, the more difficult it is to derive it from the data. Hence, the open issue is how to balance the complexity of representation and the complexity of the algorithms needed for its automatic derivation.

Earlier in this chapter, we mentioned that there are essentially two approaches to structural representation: syntactic and topological. Purely syntactic approaches to pattern recognition have been repeatedly criticised (e.g. Tanaka, 1995 and Watanabe, 1985) for making unrealistic "grammaticallity" assumptions about the data (Section 1.1.4). Just as there is no such notion as grammaticality for images, it is difficult to conceive (by looking at the real data) some notion of grammaticality for the spectrum of an acoustic waveform or the recording of articulator movements. Hence, modern structural approaches usually adopt a topological, similarity-based, approach to modelling.

The adoption of a topological approach leads to the two issues we discuss next. The first issue is how to introduce a good metric on a set of objects. Unlike regular vector spaces, there are infinitely many similarity measures which can operate on the symbolic object representations. Adoption of each of these measures results in a symbolic modelling space with unique properties[4]. The second big issue is *what* and *how* to learn in such a space during the generalisation stage. The difficulties arise because the theory of symbolic spaces is in a much less analytically developed stage than the vector space theories. Thus, on the one hand symbolic spaces seem to offer a bigger modelling freedom than the vector spaces, but on the other, severely restrict the amount of available analytical tools[5].

[4] As was mentioned earlier in Section 1.1.4.2, where the topological approach was introduced, the symbolic space is defined as a set of objects *together* with the numeric similarity measure defined on this set. Hence, the main difficulty is not with the similarity measure, but rather with the structure of the objects (which is now richer than the structure of the numeric feature vectors) which induces different similarity measures.

[5] This does not necessarily mean that new tools cannot and should not be developed.

1.3.2 Unification of Structural and Numeric Approaches

Given structural and numeric approaches to pattern recognition, one of the interesting questions is how to combine them within a single framework. Compared to other pressing issues in pattern recognition which are investigated within computational learning theory and machine learning (quality of the classifiers, model selection and combination and so on), this issue has received relatively little attention. Similar to the explanation given in the previous section, we believe this to be due to the limited attention that the notion of object and class representations received over the years. It is interesting to note that the issue of potential unification of the two approaches was raised quite early on, at the onset of the field (e.g. Aiserman, 1969). Why is such a unification desirable? As we saw from the previous discussion, each approach possesses interesting features lacking in the other. The structural approach provides an expressive modelling space for representations, while the numeric approach provides a powerful analytic environment for decision making.

In general, there are two different approaches to unification. It is interesting to note that both approaches are primarily motivated by the idea that adequate representation leads to adequate generalisation. In this respect, these approaches are representation-centric and are of primary interest to the author in the scope of this thesis. These approaches roughly correspond to two possible (and related) interpretations of the definition of what pattern recognition is. In Section 1.1, pattern recognition was (very informally) defined as the ability to perceive regularities in patterns of classes and relate them. The natural question that arises next is how to interpret the regularity? The proponents of the *similarity-based* approach believe in the primacy of the *similarity measure* on which the representations should be based. The proponents of what one can call a *class-centric* approach (this group of researchers includes the author of this thesis) believe the notion of a *class* is a central one. The theoretical argument in favour of this or the other approach is outside the scope of this thesis. The difficulty arises because either approach is believed to be subsumed by the other. Sometimes in practical applications, as will be shown later on in this work, frameworks motivated by both of the above considerations can be employed. Below we briefly overview the main tenets of these two approaches to representation. Before proceeding it should be noted that each of the approaches leads to entirely (mathematically) different modelling frameworks, despite the fact that, theoretically at least, the two approaches seem to be related.

Some of the initial ideas of the similarity-based approach appeared in Lev Goldfarb's thesis at the end of the 70's (Goldfarb, 1979), consequently extended and formalised by him five years later (Goldfarb, 1984, 1985). One of the main contributions of that

work at the time was that it laid the formal foundations of the study of similarity-based pattern recognition. The main tenet of similarity-based pattern recognition is that the object representation cannot be taken out of the context of similarity. It is the similarity measure which *induces* the modelling space, rather than the objects themselves. From a conceptual point of view, the use of dissimilarities between objects instead of objects themselves allows the formation of a bridge between numeric and symbolic approaches. This is because the notion of similarity is universal, capturing both structural and numeric features of the original objects. Hence, for instance, instead of using a feature vector or a string for representing a certain real-world object, in the similarity-based approach this object is represented by the vector consisting of similarities between this object and rest of the objects in the training set. Hence, the similarity-based representation is essentially numeric. The properties of these modelling spaces and the classifiers in these spaces are the main subject of study in this area. In the last decade, the similarity-based approach (which is sometimes called the *featureless* approach) started receiving renewed interest in the pattern recognition and machine learning community. For a recent treatment of this subject, the interested reader is referred to Bunke *et al.* (2001); Duin *et al.* (2004); Edelman (1998); Graepel *et al.* (1999); Mottl *et al.* (2002); Pękalska *et al.* (2004), and others. An excellent study of the field is provided by Pękalska's recent thesis (Pękalska, 2005)[6].

Class-centric approaches assert the primacy of structural class representation over similarity-based representation. There are several reasons for this. One of the reasons is that similarity-based vector space representations, however well mathematically grounded, still suffer from rigid mathematical structure imposed by the underlying vector space. Hence, it is argued that the similarity-based approach cannot produce class representations which are sophisticated enough to approximate the inherent, essentially morphological, complexity of the real-world classes. As a result, by adopting a class-centric approach to modelling, one starts with a structural, usually topological (Section 1.1.4.1), representation. The essential difference between this approach and the conventional topological approaches to structural representation is that the class-centric framework has to have certain analytic means of deriving non-trivial structural class representations during the process of generalisation. Furthermore, the similarity is *induced* by the class structure (in comparison with the similarity-based approach, where the similarity measure is seen as something external to the framework). The latter assumption is motivated by experimental evidence from cognitive science. It has

[6] Although from a technical point of view there is a difference between the notions of similarity (proximity) and dissimilarity, in this thesis we use these notions interchangeably. In general, we assume that the various metrics define similarity measures, i.e. the more similar the objects being compared, the smaller the measure.

been observed that humans are not only able to *identify* a concept to which a certain perceived object belongs, but are also able to justify the latter decision by *describing* the concept (class) in terms of its attributes (Abela, 2001).

The approach we undertook in this thesis for structural representation of speech is class-centric. The following sections are devoted to introduction of the main ideas of this approach.

1.3.3 Topological Class-Centric Approach to Speech Representation

In this section we describe the motivation behind the development of a class-centric approach to representation based on topological principles. The adjective "topological" refers to the fact the modelling is (mostly) structural, but it is guided by a similarity measure. This approach incorporates several techniques described below.

Any attempt at modelling must start somewhere. An obvious point of departure for any structural representation is the choice of the objects being modelled. As was mentioned in Section 1.1.4.1, the essence of the structural topological approach is the choice of the set of objects, together with some similarity measure defined on those objects. The similarity measure is defined in a way that reflects the morphological makeup of the objects. When modelling speech, it is clear that the structure of the objects has to be somehow detected in the data. Without going into detail (a concrete topological representation of speech "objects" is described in Chapter 2), convenient units that can be extracted from speech (e.g., as shown by King and Taylor, 2000) correspond to the atomic units of linguistic analysis — phonological distinctive features. Hence, instead of feature vectors, a frame-based topological representation can be based on the symbolic bundles of phonological distinctive features. Any linguistic object (e.g. phone, syllable) can then be explicitly represented in terms of this structure. Next, a similarity measure is defined on the set of objects. Structural similarity measures usually take into account the internal *atomic* structure of the objects being compared. For instance, conventional transformation distances calculate the number of atomic operations needed for the objects to become identical.

How does the above object representation relate to a concept of class? An elegant extension of the above, called the *Evolving Transformation System* (ETS), was proposed by Goldfarb in his early papers (Goldfarb, 1990, 1992). The central idea of the proposed approach (which we denote ETS_0) is the notion of *transformation*. In its simplest form, a transformation corresponds to the basic operation employed by the transformation distances mentioned above. One can think about representation of a certain class of objects as a set of transformations. Initially, class representation consists of the basic operations only. Since the basic operations are common to all classes in the training set,

such a class representation is not interesting. This "rigid" representation corresponds to the conventional topological object representation described above. What makes a given class different from the other classes is the existence (among the objects representing that class) of common structurally non-trivial attributes, or non-trivial transformations, of discriminating nature. These (weighted) transformations induce a class-specific similarity measure which better separates members of this class from all other objects. In ETS_0, given the objects belonging to a class, the goal of the learning phase of generalisation process is to discover an optimal set of non-trivial weighted transformations using some discriminating technique. At each stage of the learning, an augmented set of transformations, discovered using the search guided by the "current" similarity measure, induces a new class-specific similarity measure. Hence, the learning stage can be seen as a sequence of "evolving" topological spaces, where evolving class representations induce an evolving similarity measure. From the speech modelling point of view, this allows description of linguistic classes in terms of their non-trivial structure. This approach is pursued in Chapter 4.

Above we introduced a "rigid" topological approach to structural modelling of objects, which can be analytically augmented (ETS_0) to allow for structural class representation. From the point of view of similarity measures, the "rigid" approach corresponds to a global similarity measure for all objects in the training set. The ETS_0 approach results in similarity measures which are class-specific. With either approach, since the particular structural representation is completely transparent for the similarity measures, it is possible to embed the symbolic representation into the corresponding vector space by using the similarity-based techniques mentioned in Section 1.3.2. This is often desirable because the symbolic spaces are not suited for visualisation and lack advanced decision-theoretic techniques available in vector spaces. The embedding techniques, resulting in a similarity-based representation, are treated in Chapter 3.

1.3.4 Formal Approach to Speech Representation

It must be emphasised that ETS_0 is not a learning algorithm but rather a model. Modelling structural class description via transformations that clarify the nature of a similarity measure is a general idea that does not force any assumptions upon the symbolic objects. For instance, one can introduce ETS_0 ideas into the setting based on strings (Abela, 2001) or trees (Kamat, 1995). From an early stage in the development of the model, it became clear (Goldfarb, 1992) that the ideas embodied by ETS_0 need to be formalised within a single unifying framework for pattern representation and recognition. In what follows we give several reasons, "technical" and otherwise, for mathematical formalisation of the above ideas (more detailed arguments, most of which

are presently outside the scope of this work, are advanced by Goldfarb, 2004 in a recent paper).

The first argument in favour of a *formal* approach to structural class and object representation is a general one. It goes without saying that human perception mechanisms operate in a rather dynamic environment. The "objects" being perceived in a certain environment are not static. In many areas of pattern recognition, especially the ones where modelling of vision and speech is concerned, it is generally agreed that "static" structural representations do not constitute an accurate reflection of the reality. In speech modelling, in particular, the above understanding led to the modelling of speech as a stochastic process (Deller *et al.*, 1993). In this respect, it is interesting to analyse the notion of a "static" representation without recourse to mathematical statistics. Our current intuitive understanding is that static representations are memory-less. Thus the static view of various perceptual processes (such as speech, vision and so on) is problematic because it contradicts the very nature of basic mechanisms of mental representation within humans. This also explains why statistical approaches have been successful. The mental representations of a real-word object must somehow encode the "evolutionary" or "developmental" history of that object. The author thinks of two alternative and very informal ways to express the latter point: from a perceptual point of view, an object is essentially a constantly evolving mental process; from an "observation" point of view, the objects under investigation appear to be constantly changing ("evolving"). Hence, one of the major motivations behind the formalisation was a desire to introduce some formal machinery which would allow to integrate a concept of an "object" and a concept of its *formative history* in a single mathematical structure. The classical discrete structures were not developed for the above purposes, hence a new approach is needed.

How can one define a formative history of an object? Starting from the earliest work on structural representation (ETS_0), Goldfarb suggested that the formative history should be seen seen as a sequence of transformations. Moreover, from a generative point of view, any object can be represented as a sequence (or collection of various sequences) of constructive transformations. Schematic illustration of the latter idea is shown in Figure 1.5, where an "object" is shown observed over several time instances. The object can be though as being formed by a various possible sequences of transformations $\{\tau\}$. Representation of a class of objects can thus be thought of as a collection of class-specific transformations together with some generative mechanism for applying them. The fundamental technical difficulty in accommodating the above ideas within a formal model can be explained by the fact that conventional symbolic representations based on discrete data structures such as strings, trees, graphs and so on, cannot be used for the

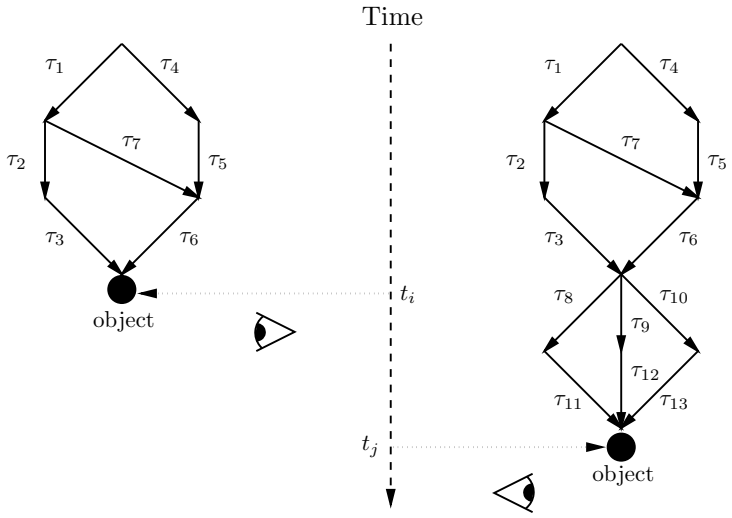


Figure 1.5: Illustration of the concept of formative history. A certain "object" is being observed with the measurements taken at times t_i and t_j. The object is shown as having several possible formative histories consisting of various transformations $\{\tau\}$, whose structure is ignored for now.

purposes of evolutionary object representation via formative histories. This is partly because when one examines a certain object, represented by a graph for instance, it is very difficult to reconstruct a sequence of transformations which constructed it. In other words, from the grammar point of view, given any object there is an exponential number of productions which can be used to generate it. In contrast with the later conventional mechanisms of structural object representation, within a formal "evolutionary" model, the representation of an object *is* its formative history. Initial formalisation of the above ideas was attempted by Goldfarb and his colleagues five years ago in (Goldfarb and Golubitsky, 2001; Goldfarb *et al.*, 2000), which resulted in a first *formal* version of ETS model we denote ETS_1.

Another important "technical" motivation for the introducing a formal framework for object and class representation can be presented as follows: given a certain object representation employing conventional structures, in order to implement ETS_0 ideas in this framework one has to employ representation-specific techniques. In theory, such approaches are rather brittle because by changing certain assumptions about the representation it is often the case that the entire suite of algorithms developed to support ETS_0 in that framework has to be completely changed. As an example, in this thesis the first chapters rely on a certain arrangement of phonological tiers of distinctive features employing strings as the basic sub-structures. Changing the representation to employ another variety of distinctive feature theory, which is based on more complex data structures, like graphs, will involve completely scrapping the algorithms developed

for string-based representation and developing new ones to support graphs. Hence, rather than focusing on the representational issues, more often than not, most of the modelling effort is spent on the issues that are not relevant to representation. Ideally, introduction of a representational formalism allows the researcher to focus on the modelling issues: derivation of atomic units of representation from the data, specification of their particular interaction, study of the class representation, and so on. The formalism, in turn, supplies formal analytic machinery for achieving the above goals, including the framework for generalisation. Over the last five years, the development of the ETS formalism has gradually been moving towards this goal, especially with the appearance of the most recent versions of the formalism (Goldfarb *et al.*, 2005a, 2004), denoted ETS_2 and ETS_4. In this thesis, for example, we describe two representations of speech developed from entirely different perspectives (production and perception-based), yet employing the same formal mathematical language. Furthermore, experiences with developing specific applications within ETS have also been instrumental in driving the development of the formal language itself.

Finally, another important motivation which drove the development of a formal approach was the desire to model the *multi-levelled* nature of mental representations. In very informal terms, any (non-trivial) transformation observed at the sensory level, becomes a primitive building block of object representation on the next level of representation. The new level of representation appears with the detection of the first transformation at a current level. Representation of a class, at any given level, is essentially a collection of transformations. Hence, class representation is also hierarchical since any primitive transformation at any given level can be "opened up" to some non-trivial structure at the previous level. A simplified two-level representation hierarchy is shown in Figure 1.6. When thinking about speech communication, our very informal intuition about this process is based on the multi-level representation hierarchy: the sender of the intended linguistic message has some mental multi-level representation which is "collapsed" and encoded as an acoustic waveform (or some other representation) and transmitted to the receiver of the message, who tries to "reassemble" the mental representation by growing it from the encoded message. More precisely, the temporal process of decoding the message involves dynamically "updating" the multi-level representation of the receiver, "growing" it only when necessary.

1.3.5 Brief ETS Literature Overview

In Section 1.3.3 it was mentioned that the origins of ETS are in the topological class-centric approach. This approach evolved from pure numeric similarity-based object representations in pattern recognition (Section 1.3.2). As mentioned above, the initial

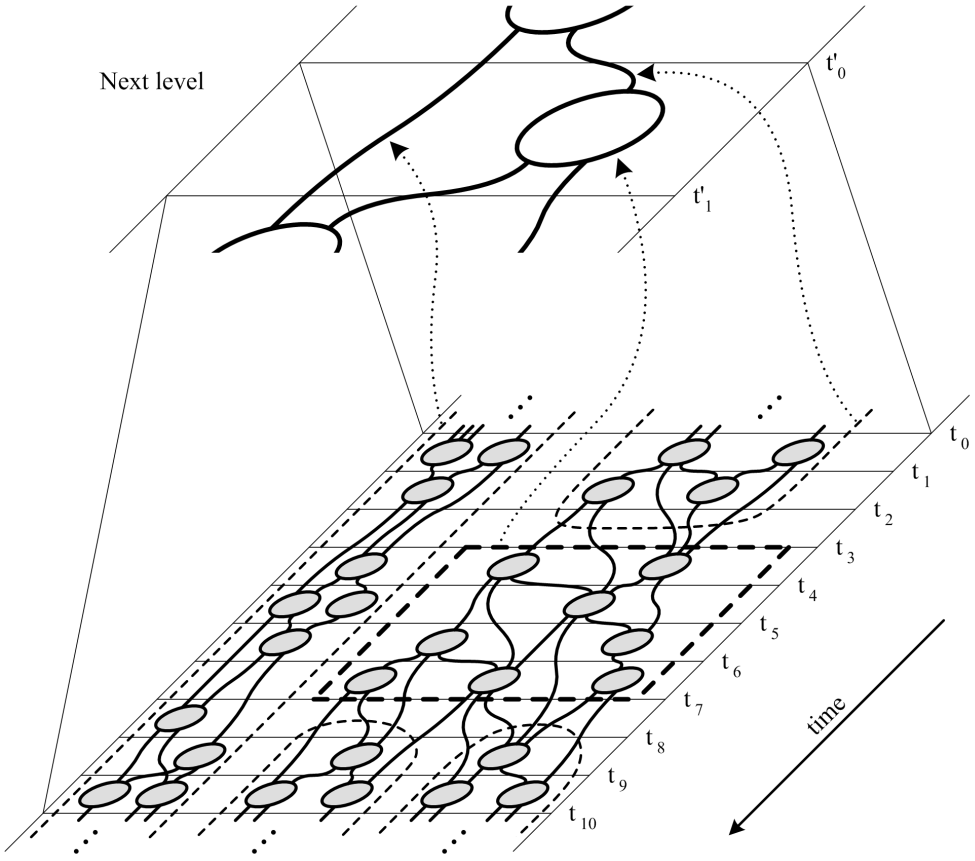


Figure 1.6: Simplified illustration of a two-level ETS_4 representation. The heavy dashed line identifies the first level transformation, consisting of atomic (primitive) transformations. On the next level, this transformation corresponds to a next-level primitive transformation (reproduced with permission from Goldfarb *et al.*, 2005a).

version of ETS, which we refer to as ETS_0, was originally proposed by Goldfarb in the early nineties (Goldfarb, 1990, 1992). Transformation-based class and object representation was developed in a string (Goldfarb and Nigam, 1994) and tree-based (Kamat, 1995) pattern recognition setting. Several position papers, clarifying some of the main assumptions of the model, appeared later on (Goldfarb *et al.*, 1995; Goldfarb and Deshpande, 1997). The most comprehensive experimental study of ETS_0 was recently completed by Abela (2001), who used this version of ETS to develop a grammatical inference system based on the ideas in (Goldfarb and Nigam, 1994). Several interesting ideas about the possible application of the model to computer vision appeared in (Goldfarb *et al.*, 1996).

After several years of gestation, the initial version of formal language, denoted ETS_1, appeared five years ago. Among the features of the ETS_1 formalism was the introduction of a unifying set-theoretic object representation with a support for formative histories and initial support for multi-level representation (Goldfarb and Golubitsky, 2001; Goldfarb *et al.*, 2000). The generative mechanism employed by the model makes use of stochastic Markov processes by attaching numeric weights to transformations. Recently, Korkin (2003) completed ETS_1 work on molecular representations in chemioinformatics. Golubitsky (2004a) explored the formal algebraic properties of the model, comparing it to "classical" discrete representations used in computer science (strings, graphs, etc.). He also provided proof of the Turing-completeness property of the model for the case when the real world environment is restricted to strings (Golubitsky, 2004b).

The initial ETS_1 version was then significantly modified to better suit the evolutionary or "dynamic" nature of the representation. A particular emphasis was put on improving the support for multi-level representation of information processes. This version of the formalism is denoted ETS_2 (Goldfarb *et al.*, 2004). It is used in thesis for articulatory representation of speech (Chapter 5). Finally, the experience with development of several (pilot) representations (including the work conducted by the author) motivated the introduction of several modification to ETS_2. This (current) revision of the formalism is known as ETS_4 (Goldfarb *et al.*, 2005a). The appearance of three revisions of the formalism in five years (ETS_1, ETS_2 and ETS_4) can be explained by the novelty of the adopted approach and general lack of research in *formal* methods for modelling natural processes in pattern recognition. Other currently ongoing work in ETS_4, which is relevant to this thesis, includes the development of structural representations for vision (Gay, 2005). The recent position statement arguing in favour of development of formal approaches appeared in (Goldfarb, 2004).

1.4 Research Objectives

Based on the above motivations, the main objectives of this thesis can be spelled out as follows:

Develop a linguistically-well motivated framework for structural class and object representation of speech based on topological modelling tools and investigate the feasibility of such a framework.

Using the topological approach to class and object representation, it is desirable to design a speech representation framework based on several principles described next. First, it should be possible to select a certain type of linguistic units (which in thesis are phonemes) and postulate their structure in terms of more primitive units (distinctive features) which can be realistically derived from speech. Next, it is desirable to introduce a similarity measure on object representations. Therefore, one can test the adequacy of the representation by using symbolic template matching techniques, generalised to operate on speech objects. In addition, it is useful to demonstrate the techniques for similarity-based transition from the symbolic modelling spaces to the corresponding vector spaces, where generalisation can be conducted more efficiently. Finally, it is desirable (with the help of ETS_0) to introduce generalisation procedures into the symbolic object representation for deriving structural class descriptions for various classes of linguistic units being modelled. The availability of a standard corpus of read speech (Garofolo, 1988; Garofolo *et al.*, 1993) makes it possible to compare, contrast and experimentally evaluate the above three techniques.

Design and explore formal articulatory representation of speech.

Investigate the feasibility of constructing a formal representation of speech based on articulatory (production-based) principles. Based on some theoretical premises of the theory of articulatory phonology that hypothesises the combinatorial structure of speech, investigate whether this combinatorial structure of physiological nature can be expressed within a formal ETS_2 framework by mapping the theoretical combinatorial units of analysis to the atomic units of the formal framework. Design techniques for deriving these atomic units directly (i.e. without recourse to non-linear mappings) from the speech recordings. Provide analysis of class representation within this framework, postulate the formal class description of various phonemes and experimentally evaluate the adequacy of the class structures on a corpus of articulatory recordings (Wrench, 2000; Wrench and Hardcastle, 2000).

1.5 Thesis Organisation

The contents of this thesis are organised into two parts. Research presented in Chapter 2, Chapter 3 and Chapter 4 resulted from the *topological* approach to speech representation, motivated in Section 1.3.3. The adjective "topological" refers to the fact that all of the approaches presented in these chapters involve structural modelling guided by a similarity measure. To be more precise, these modelling spaces should more appropriately be called *metric spaces* because the modelling involves structures together with similarity measures defined on them. The second part of this thesis consists of Chapter 5, where speech representations are defined within a *formal* mathematical framework which is specifically being developed for the purposes of representing the natural processes (Section 1.3.4). The conclusions are summarised in Chapter 6 and the future directions of research are presented. The more detailed outline of each chapter is given below. It is important to note that the logical sequence of the chapters corresponds to the evolution of author's ideas about the structural modelling of speech.

Chapter 2. Phonological Symbolic Metric Spaces:

This chapter introduces linguistically motivated structural *object* representation together with the similarity measure defined on objects. We refer to this structural modelling framework, which corresponds to topological approach of Section 1.1.4, as *phonological metric space*. To a certain extent, this representation corresponds to a symbolic version of the segment-based approach to speech recognition (Section 1.2.2), with the main difference that the segments are interpretable. The objects under investigation correspond to phones (without the loss of modelling power the objects can be extended to syllables or other larger speech units). We then explore the possible ways of introducing generalisation into this modelling framework and experimentally evaluate the discriminatory capabilities of this representation on a standard corpus of continuous speech.

Chapter 3. Pseudo-Euclidean Embedding of Phonological Metric Spaces:

As a next step further, in this chapter we investigate the *similarity-based* transition from the phonological metric space to an equivalent (generally non-Euclidean) vector space. The equivalency is defined solely on the basis on the structural similarity measure. This transition results in a similarity-based vector space representation (Section 1.3.2), where efficient analytical tools are available for learning and classification. The rationale behind this transition is simple. We investigate whether it is feasible to focus on the *object* representation in a structural modelling space and then transfer the modelling

problem into an equivalent vector space domain where the generalisation can be more efficiently approached. The properties of the resulting representation are then described, followed by an experimental evaluation.

Chapter 4. Inductive Learning with ETS_0:

In this chapter we return to the phonological metric spaces first introduced in Chapter 2 and augment the original modelling framework so that it can also discover *class* representation. In essence, this chapter introduces the first (in this thesis) *class-centric* structural approach to speech representation (Section 1.3.2). The original phonological metric space is modified so that the similarity measure defined on the objects is now not rigid, but dynamic (evolving) and is structurally induced by the class representations of the speech units (phones) in question. These modelling ideas correspond to the main tenets of the ETS_0 framework. Categorisation mechanisms in the resulting modelling space are then discussed, followed by experimental evaluation. The ETS_0 approach is compared to related similarity-based approach presented in Chapter 3.

Chapter 5. Formal Articulatory Representation of Speech with ETS_2:

This chapter presents a production-oriented approach to structural speech representation within the ETS_2 formalism (Section 1.3.4). We present the ETS_2 model and introduce an articulatory ETS_2 representation, motivated by the theory of articulatory phonology. On a technical side, we show how to derive this representation from continuous articulatory recordings and present several application-specific simplifying assumptions. The representation of several classes of phones is then experimentally evaluated on a corpus of articulatory recordings.

Chapter 6. Conclusion and Future Research:

In this chapter we outline findings and experiences with the various approaches to speech representation explored in this thesis, summarise the contributions, and present future research directions.

Reading Suggestion

Each chapter of this thesis is reasonably self-contained. The care was taken to ensure that all the necessary background material for each chapter is provided in the corresponding preliminary section, following the introduction. The author felt that such a layout is better suited for presenting the unrelated background theory of various modelling spaces considered in this thesis.

1.6 Publications and Declaration

Some of the material contained in this thesis appeared in reviewed conference and workshop proceedings during the work on this thesis. More specifically, the latter include the following: Gutkin and King (2004b) (Chapter 2), Gutkin and King (2004a) (Chapter 3), Gutkin and King (2005b) (Chapter 4), Gutkin *et al.* (2004), Gutkin and Gay (2005b), Gutkin and Gay (2005c), Gutkin and King (2005a) (Chapter 5). Author has been involved in the development of ETS_4 formalism as a member of Inductive Informatics Group run by Lev Goldfarb. His contribution to ETS_4 is indicated in (Goldfarb *et al.*, 2005a,b, Section 1.5). One paper was accepted for publication, but withdrawn by the author (Gutkin and Gay, 2005a), hence not appearing in press.

Apart from where stated otherwise (by acknowledging the work of others at the appropriate points in the text), none of the material contained in this thesis is the result of collaborative work. The length of this thesis, including footnotes and excluding the bibliography and the index, is 64,413 words.

Part I

Topological Approaches to Structural Representation

Chapter 2

Phonological Symbolic Metric Spaces

2.1 Introduction

Current automatic speech recognition (ASR) systems are usually based on Hidden Markov models (HMMs) of phonemes; speech is modelled as a linear sequence of these phonemes, like "beads on a string" (Ostendorf, 1999). Phonemes have no explicit internal structure in these systems beyond the topology of the HMMs used to model them (usually three emitting states in a simple left-to-right arrangement) (Young, 2001). The accuracy of such systems appears to have reached a plateau, motivating many researchers to look for alternative approaches. In this chapter, a novel representation and classification framework based on *symbolic structural* principles is presented. The exposition is based on our initial report in (Gutkin and King, 2004b). The method we propose is motivated by the fact that a symbolic space is well-suited for capturing and exploiting structural properties of speech which HMMs (and other models based on similar principles) fail to capitalise on.

Since speech waveforms are not symbolic, we must make a transformation into a symbolic representation. At a very low level, frame-based vector quantisation will do this, but we reject this approach since the symbol set is chosen purely on acoustic grounds. Other techniques, such as generalised feature extraction based on structure detectors (Olszewski, 2002) have been proposed recently. Structural detectors extract some very general (non-linguistic) symbolic information directly from the signal and provide it as an input to syntactic pattern recognition frameworks operating on time series data. Such approaches might potentially offer a natural way of extracting symbolic features and further research is needed to establish whether they are suitable for Automatic Speech Recognition (ASR) tasks. We have chosen to make the transition from

vector-space to symbolic representation at a linguistically well-motivated level: *phonological features*. Phonological features are a representation of speech which has several attractive properties and are better modelling units than conventional phonemes. Furthermore, it has been shown in (Frankel and King, 2005; King and Taylor, 2000) that recurrent neural networks can be successfully used to perform accurate phonological feature detection from speech signals.

Once the fundamental symbolic units of representation are obtained from the speech signal, the next issue which needs to be addressed is how to model the objects. It appears that modelling in a symbolic space is inextricably linked with the fundamental notion of a *metric space*. Briefly, a metric space is defined as a collection of *objects* in some environment *together* with a *dissimilarity measure* defined on these objects. The dissimilarity measure can take several names depending on the mathematical properties of this function, like *metric*, *pseudo-metric* and others (see Section 2.2). In this chapter, we introduce a mathematical structure for modelling which we call a *phonological metric space*. The objects of this space (*phonological templates*) are the structural models of the phonemes of speech built using symbolic phonological distinctive features. We also define several dissimilarity measures (*phonological metrics*) which operate on these objects. In particular, during the formalisation of this new modelling space, we extend several concepts and algorithms from conventional, string-based, structural pattern recognition to phonological metric spaces.

Finally, we describe experiments on a standard ASR speech classification task and compare the results with conventional numeric models. The goal of the experiments was to verify the adequacy of the proposed metric space (object representation and the metrics). In addition, we studied the behaviour of the symbolic metric-based classification and clustering algorithms, when confronted with a large symbolic dataset.

Overview of the chapter

This chapter is organised as follows. In Section 2.3 we introduce the structural object representation based on phonological feature structure. We complete the proposal of the phonological metric space in Section 2.4 by introducing the symbolic dissimilarity measures operating on the phonological objects. Various clustering and classification algorithms for the new metric space are described in Section 2.5. Section 2.6 describes the experimental setup along with the discussion of the results. We summarise the chapter in Section 2.7 and present some future directions of research which will potentially improve our metric space-based model.

2.2 Preliminaries: Metric Spaces

In this section, we briefly introduce the concept of a *metric space*, which will be extensively used in the subsequent developments. The exposition is rather informal and is based on several sources from general topology (Engelking, 1989; Khamsi and Kirk, 2001).

A metric space is an axiomatisation of the notion of closeness of points: in a metric space, to every pair of points corresponds a real number, which we treat as the distance between them. The fundamental properties of the notion of a distance are described by a following set of axioms.

Definition 2.1 (Metric Space)**.** A *metric space* is a pair (M, d) where M is a set and

$$d\colon M \times M \to \mathbb{R}^+$$

is a mapping of the Cartesian product $M \times M$ into the set of non-negative real numbers $\mathbb{R}^+$ satisfying the following axioms:

(M1) $d(x, y) = 0 \Leftrightarrow x = y$.

(M2) $d(x, y) = d(y, x), \ \forall x, y \in M$.

(M3) $d(x, y) + d(y, z) \geq d(x, z), \ \forall x, y, z \in M$.

The set M is called a *space*, the elements of M are called *points*, the function d is called a *metric* on the set M and the number $d(x, y)$ is called the *distance between x and y*. □

Condition (M1) asserts that the distance between two distinct points is positive and every point has distance zero from itself. Condition (M2) asserts that the distance is a symmetric function, not dependent on the order of points x and y. Condition ($M3$), called *triangle inequality* states that the sum of two sides of a triangle, formed by the three points, is not smaller than the third side.

The concept of a metric space derives from a more general concept of a semimetric (sometimes also called *pre-metric*) space, defined below:

Definition 2.2 (Semimetric Space)**.** A pair (M, d) only satisfying the two axioms (M1) and (M2) is called a *semimetric space*. Thus, a semimetric space is a metric space if it satisfies the triangle inequality (M3). □

An even more general metric space is defined below:

Definition 2.3 (Pseudo-Metric Space)**.** A function d defined on the set $M \times M$, assuming non-negative real values, satisfying the condition (M2) and the following condition

(M1') $d(x, x) = 0, \ \forall x \in X$

is called a *pseudo-metric* on the set X. □

As we can see, the pseudo-metric space is a space satisfying the symmetry condition (M2). In addition, it relaxes the requirement that the distance between two different objects has to be non-zero. Condition (M1') only requires the distance from any object to itself to be zero.

2.3 Phonological Object Representation

Having briefly introduced the concept of a metric space in the previous section, we are ready to consider the first core notion which was used in the that definition, namely the concept of an *object*. As was mentioned at the beginning of this chapter, we chose to represent speech on the phonological level in terms of phonemes. The phonemes, therefore have to be treated as the structured objects of the representation and our goal in this section is to provide the structural means of description for the phonemes.

In Section 2.3.1, we briefly describe the atomic structural units used for the construction of the representation. These units correspond to *phonological distinctive features*. Detection of the phonological distinctive features in continuous speech is introduced in Section 2.3.2. Since the outputs of feature detector are not symbolic, we must make a transition to a symbolic representation. This vector space to symbolic space mapping, provided by the means of quantisation, is described in Section 2.3.3. Finally, in Section 2.3.4 we introduce the symbolic objects obtained by the above steps — the *phonological feature templates*.

2.3.1 Atomic Representational Units: Phonological Features

What are the atomic units of speech representation ? For numeric models, these units are provided by the real numbers which form the foundation of vector spaces. Structural approaches, on the other hand, allow more freedom of choice when it comes to the atomic units of representation. This freedom allows one to choose units which are more abstract, and thus more expressive, than the numeric ones.

The atomic units of representation we chose consist of a set phonological distinctive features. Distinctive features are seen in various phonological theories as the atomic units fully and economically describing the phonemic inventory of any given language. In turn, a phonemic inventory (usually consisting of a few dozen categories) is used to describe the possibly unlimited range of sounds (phones or segments) encountered in spoken language. Any *phoneme* is a minimal contrastive sound unit of a language (two

phones are different phonemes if they produce phonological contrast). It is represented as a bundle of simultaneous atomic units, whose combination of properties makes a phoneme. Moreover, since the distinctive features possess well defined semantics, they are considered by many to be the basic units of linguistic analysis (Jakobson, 1978).

The ideas which led to the establishment of a Distinctive Feature Theory first appeared in the work of Trubetskoy (1958), Jakobson *et al.* (1963) and Jakobson and Halle (1971). The appearance of Transformational Generative Grammar (Chomsky, 1957), made it possible to formalise the earlier observations within the theory of Generative Phonology. In addition to providing a compact means of representing phonemes, distinctive features were shown by Chomsky and Halle (1968) to be an efficient tool for concisely representing the complex phonological processes, such as assimilation (for example, the [n] sound in "in" becomes [m] when followed by [b] in the word "in-between"). The assimilation process which transforms [n] into [m] is concisely explained by the spreading of one feature — *place of articulation* — from the [b] backwards into the [n] (King and Taylor, 2000).

There is no consensus among the abundant phonological feature theories as to what constitutes the "right" set of distinctive features. Among various feature systems one can find the binary feature system of Chomsky and Halle (1968), the Government Phonology primes of Harris (1994), Feature Geometry of Clements and Hume (1995) and many others.

Despite the differences between various feature systems, there are three common principles to which they adhere:

- The feature set should be able to characterise all the contrasting segments in human languages, preferably by use of independent and non-redundant fundamental units. In particular, this means that the feature set should be universal, not dependent on a phonemic inventory of a particular language.

- The feature set should be able to concisely and clearly describe the *natural classes*. The "naturalness" alludes to the fact that there must be some universal patterns of phonological processing among humans which are "natural" and language-independent.

- Transparency with regard to phonetic correlates. This allows the establishment of phonetic similarity by grouping the sounds by common distinctive features. Hence one can relate the behaviour of phonological processes to their corresponding phonetic (surface) realisations.

We use one of the most popular feature systems motivated by the work of Ladefoged (2001): multivalued features. Each of these features takes one of several possible values

Feature	Possible Values		
centrality	*central*	*full*	*nil*
continuant	*continuant*	*noncontinuant*	
frontback	*back*	*front*	
manner	*vowel*	*fricative*	*approximant*
	nasal	*occlusive*	
phonation	*voiced*	*unvoiced*	
place	*low*	*mid*	*high*
	labial	*coronal*	*palatal*
	corono-dental	*labio-dental*	
	velar	*glottal*	
roundness	*round*	*non-round*	
tenseness	*lax*	*tense*	

Table 2.1: The multivalued feature system. All features can additionally take the value 'silence'.

— for example, manner of articulation is one of: approximant, fricative, nasal, stop, vowel and silence. The multivalued feature system is shown in Table 2.1.

For our experiments, described at the end of this chapter, we used a smaller subset of the multivalued feature system: front-back, place of articulation, manner of articulation, roundness and voicing. The motivation for this particular choice of features is provided in (King and Taylor, 2000).

2.3.2 Detecting Distinctive Features in Continuous Speech

In this section we briefly outline the mechanism for an automatic detection of phonological distinctive features in continuous speech. Such a mechanism is necessary because it facilitates the automatic construction of the final representation. At this point in the chapter we would like to note that this step during the construction of the representation is a classifier in itself.

The automatic detection of phonological distinctive features has received significant attention in the ASR community. This interest is primarily fuelled by the apparent limitations of the traditional acoustic approaches to ASR, such as Hidden Markov Models (HMMs). In the traditional approach to ASR (see Young, 2001 and Jelinek, 1997 for an overview), speech is usually represented by a linear sequence of acoustic models, arranged like "beads on a string" (Ostendorf, 1999). Each model represents a phoneme

or short context-dependent sequence of phonemes (this arrangement is due to the fact that words in the lexicon can easily be re-written as sequences of phonemes). For a critique of this approach to modelling, see Ostendorf (1999) and King and Taylor (2000). Among the problematic issues are the following:

- The models have to take the highly context-dependent nature of phonemes into account. This is, however, computationally intractable since inclusion of the context (previous phone, for instance) leads to an exponential explosion in the number of model parameters to estimate during the learning stage.
- It is assumed that the sequence of acoustic observations, which are highly dynamic and nonlinear, can be synchronised with a linear sequence of phonemes (models).

In order to address these concerns, some researchers have argued for the use of alternative units for ASR (Ostendorf, 1999). Among the candidates for the proposed alternative units are distinctive phonological features (King and Taylor, 2000; King *et al.*, 2000). In particular, King and Taylor have shown in (King and Taylor, 2000) that recurrent neural networks can be successfully used to perform accurate phonological feature detection from speech signals (the details on the performance of the detector are given later on in this chapter, in Section 2.6.2). This detection can be viewed as a nonlinear mapping from the acoustic to the phonological space. We used their methodology in our work and the details of the experimental setup for detection of multivalued features (Gutkin and King, 2004b) are the same as those described by Wester (2003).

Let $\{f_1, \ldots, f_{N_f}\}$ denote the set of N_f multivalued features. In Section 2.3.1 we mentioned that each multivalued feature f_j, $1 \leq f_j \leq N_f$, is not binary, but can take multiple values (the number of which will be denoted by M_{f_j}). The neural networks that recover multivalued features (one neural network for each of the N_f features) from speech use a 1-of-M_{f_j} encoding on their output units. Hence there are M_{f_j} real-valued outputs (ranging from 0 to 1) for each feature f_j. The total number of such values produced by the neural network for each speech frame is

$$N = \sum_{j=1}^{N_f} M_{f_j} \,. \tag{2.1}$$

Since the training data (TIMIT corpus of read speech described by Garofolo *et al.*, 1993) is fully labelled and segmented (by human experts), it is possible to label each frame in the data with the corresponding phonological feature values. Network training is achieved by specifying canonical targets (0 or 1) for each labelled frame, but at runtime the output activation values take continuous values between 0 and 1, and the features change value asynchronously (see Figure 2.1).

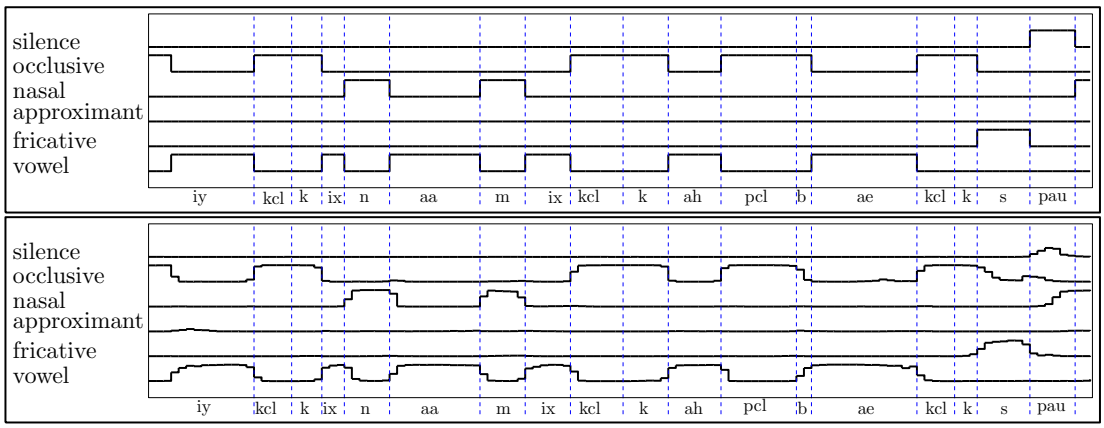


Figure 2.1: Example network output for the words "...economic cutbacks" for the manner feature of the multivalued feature system. The top plot shows the target values as derived from the canonical phoneme representation. The bottom plot shows the output of the neural net. Reproduced with permission from King and Taylor (2000).

Example 2.1. Figure 2.2 shows the phonemes and the syllables (the first two graphs on the top of the figure) of an utterance "the cat's meow" expressed by multivalued distinctive features (the three bottom graphs) automatically recovered from the TIMIT corpus of read speech by Wester (2003). Manner, voicing and place of articulation features are shown. The symbols [kcl] and [tcl] are closures corresponding to stops [k] and [t] while [h#] is a silence marker. As can be seen from Figure 2.2, the nonlinear mapping from acoustic to phonological space exhibits asynchrony, with each individual features usually not changing value directly at the phoneme or syllable boundaries. For example, the devoicing at the onset of "cat's" occurs after the left phoneme boundary of [k] (the labels are placed at the start of segments). ▷

2.3.3 Transition to Symbolic Space

During run time, the outputs of the feature detecting neural networks are not binary but continuous (ranging from zero to one) and can be interpreted as probabilities of certain features being present in the sound corresponding to the current frame. Since each probability measurement recovered in this way has a direct linguistic interpretation, we assume that this numeric measurement corresponds to a certain linguistic fact (e.g. degree of voicing) and can thus be represented symbolically, turning the neural networks into an effective structural detector.

For each of the N_f multivalued features it is therefore possible to map the M_{f_j} continuous activation values of the corresponding neural network into the symbols using simple quantisation over some finite alphabet Σ, effectively obtaining M_{f_j} independent

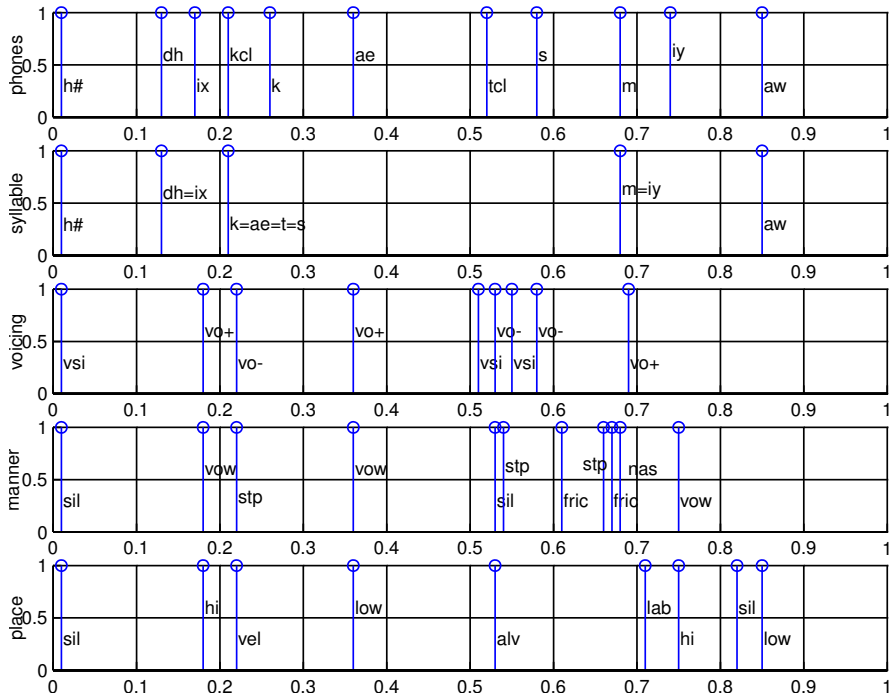


Figure 2.2: Phonemes and syllables of an utterance "the cat's meow" expressed by multivalued distinctive features. Manner, voicing and place of articulation features are shown. The symbols [kcl] and [tcl] are closures corresponding to stops [k] and [t] while [h#] is a silence marker. This figure has been produced by the software developed by Wester (2003) for her research.

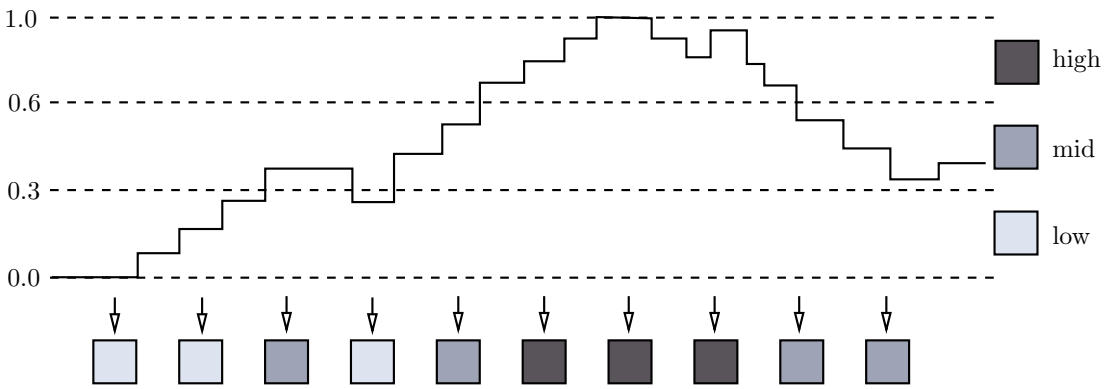


Figure 2.3: Example of a quantisation of one output activation value of a neural network over a three symbol alphabet. The resulting stream consists of 10 symbols.

time series, which we call string *streams*. The overall number of distinct streams obtained using this procedure is N, where N is given by equation (2.1). Note that each vector of N symbols thus obtained corresponds to one speech frame.

Example 2.2. Figure 2.3 shows a simple quantisation process of some continuously valued output into the string over some three symbol quantisation alphabet. The resulting symbolic stream consists of 10 symbols.

It is important to note that the names of the symbols (high, mid and low) in Figure 2.3 (and in Figure 2.5 on p. 58) denote the ranges of the quantised probability values rather than the values of the distinctive phonological feature *place of articulation* from Table 2.1 (p. 53). ▷

2.3.4 Phonological Templates

The speech has now been transformed into a sequence of vectors of symbols (N symbols for each frame). Given the phonemic boundaries, the speech can now be seen as a linear sequence of *symbolic matrices*, each identifying a phoneme in terms of its distinctive phonological features. We are now ready to introduce the most fundamental element of the representation — the objects. In what follows, we shall explain how these objects can be defined via the symbolic matrices and also discuss the structure of these objects in more detail.

Our phoneme representation system is based on the objects which we call *phonological templates* (or simply templates). Each phoneme class P is a set $\{p\}$ of one or more templates. Each template p is a realisation of class P, encountered in the data. It is convenient to represent a template as a matrix of symbols. A template p of class P, $p \in P$, is shown in a matrix format in Figure 2.4, where t_p is the start time, k_p

is the duration of p in frames and N is the fixed number of distinctive phonological feature-values given by equation (2.1).

$$\begin{matrix} f_1^{t_p} & f_1^{t_p+1} & \dots & f_1^{t_p+k_p-1} \\ f_2^{t_p} & f_2^{t_p+1} & \dots & f_2^{t_p+k_p-1} \\ \dots & \dots & \dots & \dots \\ f_N^{t_p} & f_N^{t_p+1} & \dots & f_N^{t_p+k_p-1} \end{matrix} \rightarrow_t$$

Figure 2.4: Matrix representation of a phonological template p from class P. Duration of phoneme p is given by the number of frames k_p.

Example 2.3. Figure 2.5 shows a simple representation for the two-class problem consisting of [p] and [b] stop consonants (two instances of each), for each of which two realisations are available. In other words, a class $P_{/p/}$ of unvoiced bilabial stops [p] is represented by two templates $p^1_{/p/}$ and $p^2_{/p/}$. A class $P_{/b/}$ of voiced bilabial stops [b] is represented by another two templates $p^1_{/b/}$ and $p^2_{/b/}$.

For the ease of visualisation only, we have chosen the SPE distinctive feature system, first introduced by Chomsky and Halle (1968). The reason for doing this is because the example above is easier to visualise with the SPE feature set. It is important to note that the template-based representation, which we introduced above, can represent SPE streams as well as multi-valued feature streams.

Each template consists of three independent distinctive feature streams (over a three-symbol alphabet) from the SPE features system ([tense], [consonantal] and [sonorant]) defined in (Chomsky and Halle, 1968). The three symbols can be interpreted as feature being absent from the makeup of the phoneme (low), feature undergoing a transition (mid) and feature being present (high). ▷

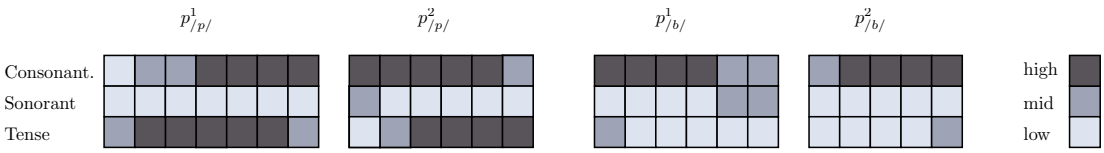


Figure 2.5: Simple three stream template representation of phonemes [p] and [b] (two instances of each) over a three symbol alphabet. The features [tense], [consonantal] and [sonorant] belong to the SPE (Chomsky and Halle, 1968) feature set.

This representation has a number of attractive features. For example, it accounts for duration. Since the durations of templates vary, even within the set representing a class (phoneme), templates of various durations can be used for a given class. Aspects

of co-articulation (such as assimilation, described above) can be accounted for, since the features are represented explicitly and independently. They can change value anywhere within a given template. Finally, this representation is amenable to human examination since its components have explicit linguistic interpretation. One of the shortcomings of this representation is, that at present, we have no way of modelling the feature spreading from one template on to the next one. This is because each template only has the knowledge of its own speech frames.

2.4 Phonological Metrics and Metric Space

Having introduced the phonemic templates, which are the structural objects in our representation, in this section we introduce the second fundamental component of the representation, without which the representation is not complete — the metric. The metric defined on the phonological templates allows us to introduce the complete representation — the metric space.

In Section 2.4.1, we define (in general terms) the metric on a set of phonological templates from the previous section. This allows us to complete the definition of the phonological metric space for our approach. We conclude the exposition in Section 2.4.2, where we give a more detailed account of the metric we use in our approach.

2.4.1 Phonological Metric Space

Once the structural representation is obtained by means of quantisation of neural network outputs, the next step is to introduce a similarity measure, defined on phonological templates.

When defining the phonological templates in Section 2.3.4 we mentioned that each template p from some class P consists of N strings of the same length k_p over a finite alphabet Σ. We called these strings *streams*.

In Section 2.2, the concept of a metric was defined to be a real-valued mapping on a set of all objects in the domain under investigation. For our phonological representation, the set of all objects is given by the set of all phonological templates which we denote by $\mathbb{P}$. The real-valued mapping on the set $\mathbb{P}$ will be denoted by $d_{\mathbb{P}}$. We proceed by defining a metric space corresponding to our structural representation:

Definition 2.4 (Phonological Metric Space)**.** A *phonological metric space* is a pair $(\mathbb{P}, d_{\mathbb{P}})$, where $\mathbb{P}$ is a set of all possible templates having N streams and

$$d_{\mathbb{P}} \colon \mathbb{P} \times \mathbb{P} \to \mathbb{R}^{+}$$

is a mapping of the Cartesian product $\mathbb{P} \times \mathbb{P}$ onto the set of non-negative real numbers $\mathbb{R}^+$, such that

$$d_{\mathbb{P}} = \sum_{i=1}^{N} d_i \,, \tag{2.2}$$

where d_i can be any chosen string similarity measure, satisfying the metric axioms from Definition 2.1. □

Remark 2.1 (Naming Convention)**.** Note that the resulting properties of the metric space are essentially dictated by the per-stream distance functions d_i. The necessary conditions for the satisfaction of metric axioms by the template metric $d_{\mathbb{P}}$ can be violated if d_i are not metric functions. In such a case, the resulting space is not a metric space. For the sake of brevity, in this chapter we will continue to refer to such spaces as metric spaces. □

The following are the important assumptions we make:

- Since it is not clear whether the use of several different metrics d_i for defining the template metric $d_{\mathbb{P}}$ in equation (2.2) can be justified on linguistic grounds, we prefer to keep the modelling simple and use the same type of metric for all the streams of any given template, i.e.

  $$\forall i \in [1, N] \colon d_i = d \,. \tag{2.3}$$

  Thus, given any two templates, p and q, the distance between them is defined by

  $$d_{\mathbb{P}}(p, q) = \sum_{i=1}^{N} d(s_i^p, s_i^q) \,,$$

  where s_i^p and s_i^q are the two strings representing stream i of p and q.

- Since the metric d, defined in equation (2.3), is common to all the string streams s_i in the template p, all the streams have to be defined over a common alphabet Σ.

2.4.2 String Metrics

In the previous section, we have defined the metric $d_{\mathbb{P}}$ operating on the set of phonological templates $\mathbb{P}$ in terms of the constituent metric d. Since each stream is a string, metric d has to be a string metric. In this section we introduce the metrics we use in our work.

In general, string metrics are usually defined in terms of edit costs. A *string edit distance*, introduces a set of edit operations with costs associated with each edit operation

and defines the distance between the two given strings as a minimum (possibly weighted) cost sequence of edit operations needed to transfer one string into another. Algorithms developed for this formalism use either character-based or block-based operations. In our work we use the more popular character edit distance algorithms. Several principles underlying the operation of string edit distances are introduced in Section 2.4.2.1, along with some important definitions.

The first, and by far the most popular, *character edit distance* algorithm was proposed by Levenshtein (Levenshtein, 1966; Sankoff and Kruskal, 1983) and improved by Wagner and Fisher who, in their classical paper (Wagner and Fisher, 1974), suggested an algorithm based on dynamic programming which achieves a time complexity of $O(n \cdot m)$, where n and m are lengths of the two strings to be compared. This algorithm has been used for decades in many applications. For example, in bioinformatics this algorithm was used for measuring the similarity of DNA and protein sequences (Needleman and Wunsch, 1970; Smith and Waterman, 1981). We briefly introduce this algorithm in Section 2.4.2.2.

An alternative metric, called the *normalised edit distance*, has been proposed by Marzal and Vidal (1993). They argued that the normalised edit distance is better suited for pattern recognition tasks than the classical edit distance. Based on their findings we decided to use this metric in our work as well. We describe this algorithm in Section 2.4.2.3.

2.4.2.1 Preliminaries

Given the two strings

$$A = a_1, a_2, \ldots, a_n \quad \text{and} \quad B = b_1, b_2, \ldots, b_m$$

over some finite alphabet Σ, the aim is to compute the edit distance between A and B. An additional symbol, not belonging to an alphabet Σ, is an empty symbol (or empty string) denoted by ϵ.

Definition 2.5 (Edit Operation)**.** An *edit operation* is an ordered pair $(c_i, c_j) \neq (\epsilon, \epsilon)$, where $c_i, c_j \in \Sigma \cup \{\epsilon\}$. String B results from string A via (c_i, c_j) if

$$A = S_1 c_i S_2 \quad \text{and} \quad B = S_1 c_j S_2$$

for some strings S_1 and S_2 over Σ. The pair (c_i, c_j) is called a *replacement* if $c_i \neq \epsilon, c_j \neq \epsilon$, a *deletion* if $c_j = \epsilon$ and an *insertion* if $c_i = \epsilon$. □

Definition 2.6 (Edit Sequence)**.** A sequence E of edit operations is called an *edit sequence*. Let

$$E = e_1, e_2, \ldots, e_k$$

be an edit sequence. B is said to be *derivable* from A if there exists a sequence of strings $S_0, S_1, \ldots, S_k$ such that $A = S_0, B = S_k$ and for $1 \leq i \leq k$, S_i results from S_{i-1} via e_i. In the worst case scenario, B is always derivable from A via a sequence consisting of n deletions and m insertions. □

Definition 2.7 (Edit Cost Function)**.** A *cost function* δ is a binary mapping assigning a non-negative real number to each edit operation (c_i, c_j). Thus, the cost of a sequence E of length k is given by

$$\delta(E) = \sum_{i=1}^{k} \delta(e_i)\,. \qquad \square$$

2.4.2.2 Wagner-Fisher Algorithm

Based on the definitions from the previous section, we can now define the weighted edit distance between the strings A and B:

Definition 2.8. The *edit distance* δ between the strings A and B is given by

$$\delta(A, B) = \min\{\ \delta(E)\ \mid\ B\ \text{derives from}\ A\ \text{via}\ E\ \}\,. \qquad \square$$

An efficient algorithm proposed by Wagner and Fisher for calculating the above weighted edit distance proceeds as follows: Let

$$A(i,j) = a_i, a_{i+1}, \ldots, a_j \quad \text{and} \quad B(i,j) = b_i, b_{i+1}, \ldots, b_j$$

denote the two substrings of A and B. Also let

$$A_i = a_1, a_2, \ldots, a_i\,,\ \ B_j = b_1, b_2, \ldots, b_j\,,\ \ \delta_{i,j} = \delta(A_i, B_j)\,.$$

Construct a $(n+1) \times (m+1)$ matrix

$$D = (d_{i,j}) \quad i \in [0, n],\ j \in [0, m]\,.$$

The first row and the first column of the matrix D are given by

$$d_{0,0} = 0\,,\ \ d_{0,j} = \delta(\epsilon, B_j) = \sum_{k=1}^{j} \delta(\epsilon, b_k)\,,\ \ d_{i,0} = \delta(A_i, \epsilon) = \sum_{k=1}^{i} \delta(a_k, \epsilon)$$

and all the other elements of the matrix D are given by

$$d_{i,j} = \delta_{i,j}\,.$$

Wagner and Fisher (1974) proved the following recursive relation:

Theorem 2.1 (Recursive Relation)**.** *At each iteration of the computation,*

$$\begin{aligned}\delta_{i,j} = \min\big(&d_{i-1,j-1} + \delta(a_i, b_j)\,,\\ &d_{i-1,j} + \delta(a_i, \epsilon)\,,\\ &d_{i,j-1} + \delta(\epsilon, b_j)\big)\,,\end{aligned}$$

where $i \in [1, n]$*,* $j \in [1, m]$*.*

Proof. See (Wagner and Fisher, 1974). □

At the last iteration of the algorithm, the edit distance between the two strings is given by $\delta(A, B) = d_{n,m}$. The algorithm uses $O(n \cdot m)$ elementary steps and $O(n \cdot m)$ space.

An open area of research deals with attempts to lower the worst case quadratic bound of this algorithm. The algorithm by Masek and Patterson (1980; 1983) achieves the best known bound of $O(m \cdot n/\log n)$. Additional improvements in time complexity have been obtained by Cole and Hariharan (1998); Landau and Vishkin (1986); Myers (1986). However, Masek and Patterson's worst case bound has not been surpassed.

2.4.2.3 Normalised Edit Distance

An alternative metric we use is the *normalised edit distance* proposed by Marzal and Vidal (1993). The normalised edit distance between the two strings A and B is defined as the minimum quotient between the sum of weights of all the edit operations required to transform A into B and the length of the editing sequence corresponding to these operations. Stated as an optimisation problem (Vidal *et al.*, 1995), the computation of the normalised edit distance is defined as

$$\delta(A, B) = \min_{E \in \mathbb{E}} \frac{\delta(E)}{|E|}\,,$$

where $\delta(E)$ is the cost of the editing sequence E from Definition 2.7, $|E|$ is the length of the editing sequence and $\mathbb{E}$ is the set of all possible edit sequences between A and B.

A straightforward procedure for computing $\delta(A, B)$ would require expanding all the possible editing sequences between A and B and computing the corresponding normalised weights. This approach would require an exponential computing time. Instead, Marzal and Vidal (1993) proposed the following efficient procedure employing the construction of the following $(n+1) \times (m+1) \times (n+m+1)$ matrix D. Let

$$D = (d_{i,j,k}) \quad i \in [0, n],\ j \in [0, m],\ k \in [0, n+m]\,.$$

Theorem 2.2 (Recursive Relation)**.** *Let n and m be the lengths of the strings A and B to be compared. Then*

$$\forall i, j, k \quad 1 \leq i \leq n,\, 1 \leq j \leq m,\, 1 \leq k \leq n+m\colon$$

1. *If* $\max(i,j) \leq k \leq i+j$ *then*

$$\begin{aligned} d_{i,j,k} = \min\big(&d_{i-1,j,k-1} + \delta(a_i, \epsilon)\,, \\ &d_{i,j-1,k-1} + \delta(\epsilon, b_j)\,, \\ &d_{i-1,j-1,k-1} + \delta(a_i, b_j)\big)\,; \end{aligned}$$

2. *Otherwise, if* $k < \max(i,j)$ *or* $k > i+j$ *then* $d_{i,j,k} = \infty$.

Proof. See (Marzal and Vidal, 1993, Theorem 4.2). □

The other values are calculated as follows:

Theorem 2.3. *The following equalities are satisfied for the rest of the entries of matrix D:*

1. $\forall i,\, 1 \leq i \leq n\colon$

$$d_{i,0,i} = \sum_{l=1}^{i} \delta(a_l, \epsilon) \quad \textit{and} \quad \forall k \neq i\colon d_{i,0,k} = \infty\,;$$

2. $\forall j,\, 1 \leq j \leq m\colon$

$$d_{0,j,j} = \sum_{l=1}^{j} \delta(\epsilon, b_l) \quad \textit{and} \quad \forall k \neq j\colon d_{0,j,k} = \infty\,.$$

Proof. See (Marzal and Vidal, 1993, Theorem 4.3). □

Having recursively populated the matrix D using the relations from Theorems 2.2 and 2.3, the normalised edit distance is given by the following equation (Marzal and Vidal, 1993, Theorem 4.1):

$$\delta(A,B) = \min_{\max(n,m) \leq k \leq n+m} \frac{d_{n,m,k}}{k}\,.$$

The algorithm described above has a time complexity of $O(m \cdot n \cdot (n+m))$. The pseudocode for this algorithm is shown in Figure 2.6.

A faster version of this algorithm was proposed by the same authors in (Vidal *et al.*, 1995) and further improved by Arslan and Egeciognu (2000) for the cases when the cost function is uniform (i.e. when the costs are associated with the type of the operation and do not depend on a particular symbol). The normalised edit distance was shown to outperform its un-normalised counterpart (regular string edit distance from the previous section) on several small pattern recognition tasks, including hand-written digit (Marzal and Vidal, 1993) and chromosome (Martínez-Hinarejos *et al.*, 2003) recognition.

NORMALISED EDIT DISTANCE(A, B)

$\mathrm{N} \leftarrow |A|$; $\mathrm{M} \leftarrow |B|$
D is an array of dimension $(\mathrm{N}+1, \mathrm{M}+1, \mathrm{N}+\mathrm{M}+1)$.
$D[0,0,0] \leftarrow 0$; $D[0,0,1] \leftarrow \infty$;
for $j \leftarrow 1$ **to** M **do**
    $D[0, j, j-1] \leftarrow \infty$; $D[0, j, j+1] \leftarrow \infty$
    $D[0, j, j] \leftarrow D[0, j-1, j-1] + \delta(\epsilon, B[j])$
for $i \leftarrow 1$ **to** N **do**
    $D[i, 0, i-1] \leftarrow \infty$; $D[i, 0, i+1] \leftarrow \infty$
    $D[i, 0, i] \leftarrow D[i-1, 0, i-1] + \delta(A[i], \epsilon)$
    for $j \leftarrow 1$ **to** M **do**
        $D[i, j, \max(i, j) - 1] \leftarrow \infty$
        for $k \leftarrow \max(i, j)$ **to** $i + j$ **do**
            $D[i, j, k] \leftarrow \min\big($
                $D[i-1, j, k-1] + \delta(A[i], \epsilon)$,
                $D[i, j-1, j-1] + \delta(\epsilon, B[j])$,
                $D[i-1, j-1, k-1] + \delta(A[i], B[j])\big)$
        $D[i, j, i+j+1] \leftarrow \infty$
$d \leftarrow \infty$
for $k \leftarrow \mathrm{N}$ **to** $\mathrm{N}+\mathrm{M}$ **do**
    $d \leftarrow \min\big(d, \frac{D[\mathrm{N},\mathrm{M},k]}{k}\big)$
return d

Figure 2.6: Pseudocode for computing the Normalised Edit Distance (Marzal and Vidal, 1993) between the two strings A and B.

2.5 Prototype Selection and Classification

Since the phonological metric space $(\mathbb{P}, d_{\mathbb{P}})$ under investigation possesses a specific template-based structure, in this section we generalise several techniques to operate in the phonological metric space and address the two issues: the efficient selection of templates by clustering and classification.

For each phoneme (class) in the training and test sets, the set of phonological templates derived from the speech signal is usually large. Since the symbolic metrics operating on the objects in question are much slower than their numeric counterparts, it is therefore desirable to have a clustering procedure for selecting a small set of the most typical (representative) members of any given class. In order to introduce the clustering procedure, we need to properly define the concept of a *mean* of a set of phonological templates. In Section 2.5.1 we introduce this notion via the generalisation of the con-

cept of a mean for the set of strings and provide two different algorithms for computing the most typical phonological template with respect to a given set. The clustering procedure is then introduced in Sections 2.5.2 and 2.5.3.

In Section 2.5.4, we address the issue of supervised classification in the phonological metric space and briefly introduce the generalisation of the efficient symbolic version of the popular k Nearest Neighbour classification rule.

2.5.1 Template Means and Medians

In what follows, we distinguish between the notions of a *mean* and a *median*. The median is an object which belongs to a supplied set. The mean is an object produced by a non-trivial construction procedure and does not necessarily belong to a set. The mean can be thought of as a generalisation of a median. This distinction is necessary to avoid the confusion between these two notions in the symbolic setting.

2.5.1.1 From String to Template Medians

Given the set of strings and a distance defined on this set, the most obvious choice for the most typical element is a *median string*, which is defined as an element of a set with a minimal sum of (possibly squared) distances to all other elements. Consequently, we define a median for the set of phonological templates by a trivial generalisation of a notion of median string as follows:

Definition 2.9 (Median Phonological Template)**.** Given a metric space $(\mathbb{P}, d_{\mathbb{P}})$ and a set P, $P \subset \mathbb{P}$, the median phonological template $\overline{p}_s$ is the member of the set P that is defined as

$$\overline{p} = \arg\min_{p \in P} \sum_{q \in P} d_{\mathbb{P}}(p, q)\,,$$

where $d_{\mathbb{P}}$ is the template distance from Definition 2.4. □

The time complexity of this algorithm is $O(N \cdot |P|^2 \cdot L^2)$ where

$$L = \max_{p \in P} |p|$$

is the maximum length (duration) found among all the templates in a set P, $|P|$ is its cardinality and N is a fixed number of streams in any given template p.

Note, that the notion of a median string breaks down for very small sets. For example, given a set consisting of only two strings, it is not possible to decide which one of the two strings is more representative of the set (Martínez-Hinarejos *et al.*, 2003).

2.5.1.2 From String to Template Means

An alternative to string median is the *string mean* (Kohonen, 1985). The mean string is a string minimising the sum of distances to each string of the set, but that does not necessarily itself belong to the set.

Definition 2.10 (Mean String)**.** Given a set of strings S over a finite alphabet Σ and some string metric d defined on this set, the mean (or generalised median) string of a set S is defined as

$$\overline{s} = \arg\min_{x \in \Sigma^*} \sum_{y \in S} d(x, y)\,. \qquad \square$$

The search for the mean string is NP-hard and in the worst case scenario, no efficient algorithm can be devised (de la Higuera and Casacuberta, 2000; Martínez-Hinarejos *et al.*, 2003; Nicolas and Rivals, 2003). However, efficient techniques for computing an approximation exist (Fischer and Zell, 2000; Kohonen, 1985). We adopted a greedy algorithm proposed by Casacuberta and de Antonio (1997). This algorithm is shown in Figure 2.7 (p. 77). The algorithm constructs the mean string symbol by symbol, making use of the dynamic programming approach for computing the Levenshtein distance. The time complexity of this algorithm is $O(K^2 \cdot |\Sigma| \cdot |S|)$, where K is the length of the biggest string in S. This algorithm has recently been further improved by Martínez-Hinarejos *et al.* (2003).

We treat the streams independently. Hence, given a set of templates P, we can find N string means independently, one for each of the N sets of streams comprising the set P. We then define a *mean phonological template* for a set P by constructing a template which consists of the discovered N string means.

2.5.2 Clustering Algorithms

Given a phonological metric space $(\mathbb{P}, d_{\mathbb{P}})$, a finite set of templates $P \subset \mathbb{P}$ and a positive integer k, the goal is to organise (cluster) the templates (in some optimal or suboptimal way) into a set Q of k clusters based on their dissimilarity $d_{\mathbb{P}}$. This is often referred to as *partitional clustering* (Jain *et al.*, 1999), as opposed to hierarchical clustering that produces a nested series of partitions based on some dissimilarity-based criterion for merging or splitting clusters.

Perhaps the most widely used partitional clustering algorithm is k-means (also known as the *basic ISODATA* algorithm) (Duda *et al.*, 2001; Jain and Dubes, 1988). In a vector space (numeric) setting, the basic structure of the k-means algorithm is as follows:

1. Initialisation: The traditional approach is to randomly generate k clusters and determine the cluster centres (*centroids*) or directly generate k seed points as

cluster centres. The centroid is the point generated by computing the arithmetic mean for each dimension separately for all the points in the cluster.

2. Assign each point to the nearest cluster centre.

3. Recompute the new cluster centres.

4. Repeat until some convergence criterion is met (usually, the algorithm is considered to have converged if the cluster assignments have not changed from the previous iteration).

Despite maximising inter-cluster (or minimising intra-cluster) variance, the algorithm is suboptimal because it can converge on a local minima of variance. The main advantages of this algorithm (despite its sub-optimality) are its simplicity and speed, which allows it to run on large datasets. A special issue, addressed in the next section, is the issue of the initial cluster assignment. Changing the initialisation strategy usually affects the final partition.

The algorithm can be generalised to operate in the symbolic space $(\mathbb{P}, d_{\mathbb{P}})$ if instead of the familiar numeric means (vectors), any symbolic equivalents of means are used. In particular, in our work we used the mean and median templates from Section 2.5.1 to represent the cluster centroids. In what follows, we refer to this generalised version of the algorithm as *k-medians*.

It is important to note that by generalising the problem to symbolic spaces, one is faced with the apparent increase in the computational complexity of the problem due to the inherent complexity of the modelling space (for example, see NP-completeness issues with regard to computing the mean template, Section 2.5.1). As we have seen, however, several computable approximations exist.

2.5.3 Clustering Initialisation Criteria

One of the peculiarities of the k-medians algorithm is that it is sensitive to the initial assignment of the clusters. Various assignment strategies result in different final cluster partitions. In this section we describe the two initialisation techniques we used in our work. The first is the generalisation of the well-known symbolic version of the *MaxMin* algorithm. The second is the initialisation technique we think is more suitable for modelling the phonological templates - the *Duration*-based algorithm.

2.5.3.1 *MaxMin* Initialisation

In a comprehensive study, Juan and Vidal (2000a) compared four different initialisation techniques for the k-medians algorithm for strings and favoured the generalised

symbolic version of an efficient initialisation technique called *MaxMin* (Tou and Gonzalez, 1974). This initialisation algorithm iteratively selects one cluster centroid at a time. At each iteration i, $1 < i \leq k$, the set Q consisting of $i-1$ previously chosen centroids is augmented with the centroid whose distance to its closest representative is maximum (Juan and Vidal, 2000a), i.e.

$$Q_i = \begin{cases} \text{rand}(P) & \text{if } i = 1\,, \\ Q_{i-1} \cup \{q_i\} & \text{if } i > 1 \end{cases}$$

where the operator rand(P) selects an arbitrary element of a set P and

$$q_i = \underset{p \in P \setminus Q_{i-1}}{\arg\max} \min_{q \in Q_{i-1}} d_{\mathbb{P}}(p, q)\,.$$

This algorithm performs approximately $n(n-k)$ distance computations. The pseudocode for this algorithm is shown in Figure 2.8 (p. 78).

2.5.3.2 Duration-based Initialisation

An alternative initialisation technique we investigated uses the *duration* of the training templates. Given the training set of size M, the templates are first sorted by duration and the data is then divided into k subsets, each containing M/k training templates with the centroids of these k subsets chosen as initial centroids.

This initialisation technique tries to account for the (possibly) high variance in the durations of the templates belonging to a set P to be clustered. Hence, the clusters are initially grouped according to their duration.

2.5.4 Classification

Perhaps the most popular classification technique is the k Nearest Neighbour (k-NN) classification rule (Duda *et al.*, 2001). In general, the k-NN classifiers label an unknown sample with the label of the majority of the nearest (with a smallest distance) neighbours. One of the most attractive properties of this algorithm to us seems to be its generality. The k-NN classification rule is entirely independent of objects in the symbolic space. Classification of the unknown objects is based solely on the basis of the symbolic space dissimilarity measure.

Given a phonological metric space $(\mathbb{P}, d_{\mathbb{P}})$, a finite set of templates $P \subset \mathbb{P}$, a test template $x \in P$ and a positive integer k, the goal is to compute an ordered list of k-nearest templates $P^* \in P^k$ (and the corresponding distances $D^* \in \mathbb{R}^k$) to the test template x.

Similar to clustering in symbolic spaces, a crucial issue which needs to be taken into account is the issue of the computational complexity of the k-NN algorithm in the

symbolic space. For large training and test sets, calculation of dissimilarity metrics may become computationally prohibitive due to the high complexity of the symbolic metric at hand. Perhaps the fastest k-NN search algorithm designed to cope with this problem is the k-Approximating and Eliminating Search Algorithm, proposed by Juan *et al.* (1998). Experiments conducted by the them on a chromosome recognition task showed that the number of distances computed during the search phase was very small and tended to be independent of the number of objects in the training set.

The pseudocode for the k-Approximating and Eliminating Search Algorithm (k-AESA) operating in a phonological metric space $(\mathbb{P}, d_{\mathbb{P}})$ is shown in Figure 2.9 (p. 79). Briefly, the templates in the training set are divided into three sets (Juan and Vidal, 2000a): selected (S), active (A) and eliminated (E), though only the set A is maintained by the algorithm. Prototypes in S are those which have already been selected to compute their distances to the test template and build the current solution. The rest of the templates are assigned to A or E in accordance to the following lower bound function for the distance from a candidate (unselected) template $\overline{p}$ to the test template:

$$g_S(\overline{p}) = \max_{p \in S} |d_{\mathbb{P}}(\overline{p}, p) - d_{\mathbb{P}}(p, x)| \,.$$

Candidate templates whose associated lower bounds are smaller than the current k smallest distance are assigned to A, while the others are eliminated from the search (by including them into E).

2.6 Experiments and Discussion

2.6.1 The Database

Our experiments used the TIMIT database (Garofolo, 1988; Garofolo *et al.*, 1993). The TIMIT corpus of read speech is designed to provide speech data for acoustic-phonetic studies and for the development and evaluation of automatic speech recognition systems. TIMIT contains broadband recordings of 630 speakers grouped into 8 major dialects of American English, each reading 10 phonetically rich sentences. The TIMIT corpus includes time-aligned orthographic, phonetic and word transcriptions as well as a 16-bit, 16 kHz speech waveform file for each utterance. The entire corpus is reliably transcribed at the word and surface phonetic levels. Test and training subsets, balanced for phonetic and dialectal coverage, are specified.

The standard training/test data partition is kept, with only the `sx` and `si` sentences being used, resulting in 3,696 training utterances from 462 different speakers, out of which 100 sentences were held out for cross-validation training of neural networks. The entire test set of 1,344 utterances from 168 speakers was used for the classification

Feature	% Frames Correct	K_h	K_o
manner	87.0	200	6
phonation	92.9	100	3
place	78.3	300	10
roundness	90.6	100	3
frontback	86.4	100	3

Table 2.2: Neural network architectures (given by number of hidden units K_h and output units K_o) used by Wester and frame-wise classification results for multivalued features she reported (Wester, 2003).

experiment. None of the test speakers are in the training set, and hence all the experiments are open and speaker independent. The phoneme set has been reduced to 39 phonemes as in (Lee and Hon, 1989; Wester, 2003). There are 46,869 phone labels in the test set and 129,162 phonemic labels in the training set, 176,031 labels overall. This experimental setup is similar to the ones in (King and Taylor, 2000) and (Wester, 2003).

2.6.2 Multivalued Feature Detection

The output of the feature detecting neural networks, which we use in the experiments described in the next sections of this chapter, was provided by Wester (2003). In this section we briefly mention some of the details of the feature detecting system she used.

Wester (2003) used five multivalued feature groups (manner, phonation, place, roundness and frontback) out of eight shown in Table 2.1. The architecture of the feature detecting networks used by Wester is essentially similar to the one used in (King and Taylor, 2000; King *et al.*, 2000), with the exception of the number of hidden (K_h) and output units (K_o) for each feature (see Table 2.2).

Frame duration of 25 ms was used, with a frame shift of 10 ms. For each 25 ms frame, the feature vector consisted of 12 Mel frequency cepstral coefficients (MFCC) plus energy. Additional components of the feature vector included delta and acceleration (delta-delta) coefficients, forming a feature vector with an overall dimension of 39. These feature vectors served as input (together with the context frames) to the neural networks. Overall multivalued feature classification results, reported by Wester (2003), calculated in terms of the number of correctly classified frames are shown in the second column of Table 2.2. For more information on the architecture of the neural networks, the learning control parameters and the validation strategy, refer to (Wester, 2003).

2.6.3 Derivation of Phonological Templates

In order to derive symbolic phonological templates, we quantised the neural network output activations using several different quantisation levels, each inducing a new alphabet Σ. For each quantisation level, the redundant templates were removed from the resulting symbolic training and test sets. By redundant templates we mean identical templates which appear in the training and test sets as an artifact of quantisation. Table 2.3 shows the training and test set sizes obtained for several quantisation levels Σ and the corresponding percentage of the overall number of redundant templates removed from the training and test sets (with respect to the original 176,031 templates in the training and test set).

Quantisation Level	Training Set	Test Set	Redundancy (%)
3	107,284	42,061	15.1
5	117,968	42,198	9.0
7	118,433	42,214	8.7
10	124,962	42,540	4.8
15	125,151	42,554	4.7

Table 2.3: Number of templates in the training and test sets for each of the quantisation levels $|\Sigma|$ (alphabet sizes).

It can be seen from Table 2.3 that the increase in quantisation level leads to an increase in the number of unique templates in both training and test sets. As the size of the alphabet increases, the sizes of the symbolic training and test sets are expected to asymptotically reach the original value of 176,031 templates, with the measure of redundancy tending towards zero. The larger the size of the alphabet, however, the larger the "symbolic variance". As a result, symbolic modelling in the metric spaces constructed over large alphabets becomes computationally more expensive. For the experiments described below, we decided to fix the cardinality of the quantisation alphabet to 10.

We assume that all the constituent streams of each phonological template are independent, hence we are weighting them all equally. Overall, each template has $N = 25$ streams corresponding to five multivalued features (see Table 2.2) associated with it.

The next issue we need to mention is the weight scheme of the edit costs we use when computing the weighted Levenshtein (given in Section 2.4.2.2) and normalised (given in Section 2.4.2.3) edit distances. In general, given the metric space $(\mathbb{P}, d_{\mathbb{P}})$, where the phonological templates are constructed over some finite alphabet Σ, we use a *uniform* weight scheme, whereby $\forall x, y \in \Sigma$ each *insertion* cost $\delta(\epsilon, x)$ and each *deletion* cost

$\delta(x, \epsilon)$ is assigned the weight $1/|\Sigma|$ and each *substitution* operation $\delta(x, y)$ is assigned the weight $2/|\Sigma|$. The weights are independent of the respective alphabet symbols.

2.6.4 Training Set Pruning

In order to reduce the number of templates in the training sets P for each class of phonemes, we clustered these sets using the k-median clustering algorithm described in Section 2.5. In order to apply clustering and classification algorithms in the metric space $(\mathbb{P}, d_{\mathbb{P}})$, we needed to generalise the corresponding crucial concepts, such as the nature of a concept of a mean in this metric space. Table 2.4 shows the concepts and algorithms involved in the clustering in metric space $(\mathbb{P}, d_{\mathbb{P}})$, along with the corresponding notation.

Algorithm Type	Available Algorithms	Notation
Similarity	Weighted Levenshtein Edit Distance	$D_{\mathbb{P}}^{L}$
	Normalised Edit Distances	$D_{\mathbb{P}}^{N}$
Mean	Median Template	$M_{\mathbb{P}}^{S}$
	Mean Template	$M_{\mathbb{P}}^{G}$
Clustering	k-medians with Duration-based initialisation	$K_{\mathbb{P}}^{D}$
	k-medians with $MaxMin$ initialisation	$K_{\mathbb{P}}^{M}$

Table 2.4: The concepts and algorithms involved in the clustering in metric space $(\mathbb{P}, d_{\mathbb{P}})$, along with the corresponding notation. The third column (notation) contains the brief names which we use to refer to the corresponding algorithms.

In order to reduce the size of the data and obtain k templates per each training set P representing the classes in question, we used two different clustering schemes: k-medians with Duration-based initialisation ($K_{\mathbb{P}}^{D}$) and k-medians with $MaxMin$ initialisation ($K_{\mathbb{P}}^{M}$). The k-median procedure makes use of the concept of mean, therefore, for each of the clustering strategies we used two different algorithms for calculating the mean of the set of templates: the median template ($M_{\mathbb{P}}^{S}$) and the mean template $M_{\mathbb{P}}^{G}$. Moreover, for all of the above algorithms, we made use of two different similarity measures defined on templates: the weighted Levenshtein ($D_{\mathbb{P}}^{L}$) and normalised ($D_{\mathbb{P}}^{N}$) edit distances.

Using the above algorithms, we reduced the size of the training set corresponding to the quantisation alphabet with cardinality of 10, which we fixed in the previous section. This training set consists of 124,962 templates (see Table 2.3). We chose the training sets for each class to be represented by the following number of templates: 5, 10, 15,

30, 50 and 100. In what follows, the number of templates per class will be denoted $|P|$.

2.6.5 Classification

During the recognition stage, an efficient k-NN AESA search technique (see Section 2.5.4) was used throughout and simple nearest neighbour (NN) search based on the score of the top candidate (in terms of the smallest distance to the test template) in the k-best list outperformed the majority voting schemes.

Classification accuracy for the data obtained using a quantisation level of 10 is shown in Table 2.5 for various values of $|P|$ (5, 10, 15, 30, 50 and 100), which is the number of centroids per class. As can be seen from Table 2.5, the schemes using weighted Levenshtein distance outperform those using normalised edit distance. The schemes using duration-based initialisation outperform those using *MaxMin*. These two findings indicate that accounting for duration is important. Also, the schemes using median outperform the one using generalised median (mean), suggesting that the construction of the mean of a set of templates (as opposed to selecting an existing member of the set) is problematic.

$\|P\|$	5	10	15	30	50	100
$M_{\mathbb{P}}^{S}/D_{\mathbb{P}}^{L}/K_{\mathbb{P}}^{D}$	54.73	58.41	58.84	59.01	59.61	**60.26**
$M_{\mathbb{P}}^{S}/D_{\mathbb{P}}^{N}/K_{\mathbb{P}}^{D}$	47.12	54.07	55.42	56.48	56.92	**58.21**
$M_{\mathbb{P}}^{S}/D_{\mathbb{P}}^{L}/K_{\mathbb{P}}^{M}$	49.89	50.62	50.03	50.59	50.74	**54.12**
$M_{\mathbb{P}}^{G}/D_{\mathbb{P}}^{L}/K_{\mathbb{P}}^{M}$	45.71	49.72	49.08	49.66	**49.84**	49.48

Table 2.5: Phoneme classification accuracy (%) for the TIMIT database.

The best result of 60.3% obtained in our experiments is lower than the state-of-the-art phoneme classification results on the TIMIT database (39 class task, core test set) reported in the literature:

- Zahorian *et al.* (1997) experimented with binary-pair partitioned (BPP) neural network classifiers. They reported 77.0% phone classification accuracy.
- Halberstadt and Glass (1997) reported the best phone classification accuracy of 79.0% obtained using the hierarchical techniques for combining mixture diagonal Gaussian classifiers.
- Clarkson and Moreno (1999) reported the results of several experiments with Support Vector Machines (SVM). The best classification accuracy of 77.6% was obtained with the SVM employing fifth degree polynomial kernel.

- Choueiter and Glass (2005) employ a novel wavelet and filter bank framework specifically designed for phonetic classification. They report the best accuracy of 77.1%.

2.7 Summary and Potential Improvements

In this chapter we have introduced a linguistically motivated *structural approach* to continuous speech recognition based on *symbolic representation* of distinctive phonological features. The structures employing phonological distinctive features are based on templates of strings. We have shown how existing notions and algorithms over strings can be adapted to our representation by extending them to operate in a specific metric space corresponding to our problem. We have also presented the results of phoneme classification experiments.

Whilst the accuracy of the system is currently lower than those reported for state-of-the-art numeric approaches, like Support Vector Machines (Salomon *et al.*, 2002) and context-dependent Hidden Markov Models (Young, 1992), we are reasonably optimistic since:

- the structural framework we have used is both intuitive and interpretable;
- the results were obtained using standard algorithms widely used in the structural pattern recognition community, especially bioinformatics;
- experiments were conducted on a task which is considered to be hard in the structural pattern recognition community. Most of the algorithms we employ have previously only been tested on small symbolic datasets;
- the system is currently very simple and there is considerable scope for improvement.

Potential Improvements

Following is a list of several issues which need to be improved on, but otherwise are outside the scope of this thesis:

Better modelling of temporal processes:

By this we mean improving the modelling power of the framework with respect to the inherent asynchrony of the phonological features, as exhibited by assimilation and co-articulation processes which operate across the phonemic boundaries. To this end, we

note that the phonological structural representation presented in this chapter can (without loss of generality) be extended to operate over larger *syllabic*, rather than phonemic, templates. As observed by Wester (2003) and Greenberg *et al.* (2002), syllables are better suited as units of linguistic analysis for modelling the asynchrony. This extension can be achieved by syllabifying the lexicon and using the syllabic boundaries during the derivation of the phonological templates.

Such an extension will increase the sizes of the phonological templates, with each template now corresponding to a syllable. The computational burden on the framework will therefore increase. In order to address this concern, we note that the efficiency of the algorithms presented above can be improved by several means. In particular, in order to handle larger templates, the constituent streams can be compressed using the technique of *run-length coding*. Moreover, efficient similarity algorithms operating on the run-length coded structures which were developed for the case of strings (Apostolico *et al.*, 1998; Bunke and Csirik, 1995; Mäkinen *et al.*, 2003), can be extended to operate over the templates. Such an extension will include, in particular, the development of the concept of mean for the set of run-length coded templates.

Better weight schemes:

One of the assumptions we made was that the streams comprising each phonological template are independent of each other, hence having an equal weight in the metric space dissimilarity measure which was defined as a sum of individual per-stream dissimilarities. This assumption is too restrictive. While the streams may be logically independent, they are of differing importance. For example, place of articulation feature in some circumstances might be more important than roundness feature. Hence, in order for representation to improve, the weights need to be introduced which better account for linguistic importance of this or the other feature. This could potentially be done along the lines of research suggested by Kondrak (2000). In addition, the optimal weights for the edit operations could be discovered from the training data at hand, perhaps by attempting to parametrise the edit distance algorithms to use fewer parameters (edit costs), along the lines of research suggested by Oommen and Loke (1999).

APPROXIMATE MEAN STRING(S)

Input: A finite set S of strings over Σ^*.
Output: A string over Σ^*.
Auxiliary:
    $\forall x \in S$: $R_x[0\,..\,|x|, 0\,..\,1]$ ▷ score (integer array)
    $\forall x \in S$: $T_x[0\,..\,|x|, |\Sigma|$ ▷ temporary (integer array)
    $E[0\,..\,M_{max}]$ ▷ (integer array of prefix lengths)
$M \leftarrow \epsilon$
for $x \in S$ **do** ▷ initialisation
        $R_x[0,0] \leftarrow 0$
        for $i \leftarrow 1$ **to** $|x|$ **do**
            $R_x[i,0] \leftarrow i$
for $j \leftarrow \min_{x \in S}(|x|)$ **to** $\max_{x \in S}(|x|)$ **do**
        $k \leftarrow j \mod 2$; $l \leftarrow (j-1) \mod 2$; $msym \leftarrow \infty$
        for $a \in \Sigma$ **do**
            $add \leftarrow 0$
            for $x \in S$ **do**
                $T_x[0,a] \leftarrow j$; $min \leftarrow \infty$
                for $i \leftarrow 1$ **to** $|x|$ **do**
                    $T_x[i,a] \leftarrow \min($ ▷ string editing
                        $R_x[i,l] + 1,$
                        $T_x[i-1,a] + 1,$
                        $R_x[i-1,l] + \delta(x[i],a))$
                    if $min < T_x[i,a]$ **then**
                        $min \leftarrow T_x[i,a]$
                $add \leftarrow add + min$
            if $msym < add$ **then**
                $msym \leftarrow add$; $sym \leftarrow a$
        $M \leftarrow \text{append}(sym, M)$
        $E[j] \leftarrow \sum_{x \in S} T_x[|x|, sym]$
        for $x \in S$ **do**
            for $i \leftarrow 1$ **to** $|x|$ **do**
                $R_x[i,k] \leftarrow T_x[i,sym]$

return prefix of M of length $\underset{1 \le j \le M_{max}}{\arg\min}\ (E[j])$

Figure 2.7: Pseudocode for computing an Approximate Mean String (Casacuberta and de Antonio, 1997) for a set of strings S.

```
MaxMin(P, k)
 1  Output: Q ⊂ P (Q ∈ P^k)
 2  Let: n ← |P|
 3  Auxiliary: D ∈ ℝ^n
 4  Q ← ∅; D ← ∞; q ← rand(P)
 5  for i ← 1 to k do
 6          Q ← Q ∪ {q}; m ← 0;
 7          for p ∈ P \ Q do
 8                  dp_q ← d_ℙ(p, q)
 9                  if dp_q < D_p then
10                          D_p ← dp_q
11                  if D_p > m then
12                          q ← p; m ← D_p
13  return Q
```

Figure 2.8: Pseudocode for the *MaxMin* initialisation of k-medians clustering algorithm (Juan and Vidal, 2000a) in a phonological metric space $(\mathbb{P}, d_{\mathbb{P}})$. The algorithm performs $n(n-k)$ distance computations.

```
K-AESA(P, x, k)
 1  Output: P* ∈ P^k; D* ∈ ℝ^k
 2  Let: n ← |P|
 3  Preprocessing: Compute matrix of inter-template distances D^{n×n} ∈ ℝ^{n×n}
 4  Auxiliary: A ⊂ ℙ; G ⊂ ℝ^n
 5  A ← P; D* ← ∞; G ← 0; p' ← rand(A)
 6  while |A| > 0 do
 7          p ← p'; dp_x ← d_ℙ(p, x); A ← A \ {p}
 8          if dp_x < D*_k then
 9                  P*_k ← p; D*_k ← dp_x; Update P* and D*
10          g* ← ∞
11          for a ∈ A do
12                  G_a ← max(G_a, |D_{a,p} − dp_x|)
13                  if G_a ≥ D*_k then
14                          A ← A \ {a}
15                  elseif G_a < g* then
16                          p' ← a; g* ← G_a
17  return P* and D*
```

Figure 2.9: Pseudocode for the k-Approximating and Eliminating Search Algorithm (k-AESA) (Juan and Vidal, 2000b) operating in a phonological metric space $(\mathbb{P}, d_{\mathbb{P}})$.

Chapter 3

Pseudo-Euclidean Embedding of Phonological Metric Spaces

3.1 Introduction

In the previous chapter, we described a classification framework based on a structural representation of speech. In that structural framework, which we called phonological metric space, phonemes are modelled as string templates at a linguistically-well motivated level, making use of the underlying phonological feature structure.

Structural representations like this, while offering a greater representational freedom than conventional vector-space approaches, have their shortcomings. The chief being the fact that the ability to apply a wide range of analytical machinery, which is available to us in vector spaces, is lost. In some cases there exist symbolic space counterparts of well-known techniques, such as k-nearest neighbours (discussed in Section 2.5), but their computational complexity is increased by the absence of the vector space properties. For example, one of the clustering techniques from the previous chapter uses the concept of a mean, which, while trivial to compute in vector spaces, is considered to be an NP-hard problem when dealing with the set of strings over a finite alphabet. It is thus not surprising that for problems such as this, we can only expect rather complex solutions which, from a computational point of view, are unlikely to match their vector-space counterparts. Such analytical limitations of the framework motivated us to consider a theory which unifies the structural and vector-space approaches for the representation of complex spoken language data (or other tasks), on one hand providing the representational convenience of symbolic spaces and on the other allowing us to use vector space decision-theoretical tools. It is such a theory, originally proposed by Goldfarb (1979; 1984; 1985) and studied over the years by Duin *et al.* (2004); Graepel *et al.* (1999); Pękalska (2005); Pękalska *et al.* (2004) and Haasdonk (2003), that we consider

in this chapter. This chapter is partially based on our previous work (Gutkin and King, 2004a).

It is the dissimilarity measure which plays a central role in the considered approach since it has been shown that, given a pseudo-metric space, it is always possible to construct an isometric mapping onto the corresponding pseudo-Euclidean vector space. This space is a member of a class of spaces in which the inner products between vectors are not restricted to be positive. Moreover, in many cases such a construction cannot be accomplished in a classical Euclidean space (Goldfarb, 1985). A brief exposition into the theory of pseudo-Euclidean spaces is given in Section 3.2.

In general, representation of patterns via their dissimilarity is an alternative to direct feature-based representation and by constructing an isometric (i.e. distance-preserving) embedding of the original pseudo-metric space all the information contained in the training sample is preserved in the vector representation (see Hjaltason and Samet, 2003 for an overview). The issues involved in construction of the isometric embedding of the original training set, as well as dimensionality reduction of the resulting pseudo-Euclidean vector representation, are discussed in Section 3.3.

Once the vector representation of the original phonological metric space is constructed in some pseudo-Euclidean space, previously unseen objects from the test set can be represented in that pseudo-Euclidean space too. This is done by calculating the metric projection of a new object onto the vector space. Different techniques for achieving this are described in Section 3.4.

Next, we describe the phoneme classification experiments conducted in pseudo-Euclidean spaces. The pseudo-Euclidean spaces are constructed from the original structural corpus with the help of the techniques mentioned above. We perform experiments in dimensionality reduction and evaluate the performance of several classifiers on small (three phonemes) and full (39 phonemes) classification tasks. The experiments are discussed in Section 3.5. We conclude the chapter in Section 3.6 and present some of the directions for future research.

3.2 Preliminaries: Pseudo-Euclidean Vector Spaces

This section provides a brief introduction to the theory of pseudo-Euclidean vector spaces, which are the generalisation of the Euclidean spaces. This generalisation is manifest in the fact that the distance between the vectors in pseudo-Euclidean space is not necessarily measured by the Pythagorean formula.

The most convenient way to arrive at a concept of generalised distance is by using the concept of a symmetric bilinear form, introduced in Section 3.2.1. Briefly, symmetric

bilinear form Φ defined on some vector space V, allows one to calculate the generalised inner product between any two vectors in V. By fixing the symmetric bilinear form Φ of the space V, a pair (V, Φ) completely describes the dissimilarity properties of the vectors in V. Note the intimate relation of this approach to topology (Engelking, 1989; Khamsi and Kirk, 2001). Various properties of vector spaces equipped with symmetric bilinear forms have been studied in the mathematical literature (Dieudonné, 1960; Gantmacher, 1959; Greub, 1967).

Section 3.2.2 presents a brief overview of the theory of the pseudo-Euclidean spaces. The pseudo-Euclidean space is a real vector space in which the matrix of inner products corresponding to the symmetric bilinear form is no longer constrained to be positive definite, i.e. the squared norm (defined as an inner product between the vector and itself) is not constrained to be positive. The theory of pseudo-Euclidean spaces is reasonably well understood (Greub, 1967, Chapter IX). We base the following exposition on an excellent self-contained overview provided by (Goldfarb, 1985, Chapter 3), which contains a detailed treatment of this topic.

3.2.1 Symmetric Bilinear Forms

Let V be a vector space over the field of real numbers $\mathbb{R}$ (in what follows, we will simply refer to V as a real vector space). One of the fundamental mathematical notions which allows one to define metrics on the vector spaces (in other words, to "metrise" the vector spaces) is the notion of a *symmetric bilinear form* (Gantmacher, 1959), defined below.

Definition 3.1 (Symmetric Bilinear Form)**.** A *symmetric bilinear form on* V is a mapping

$$\Phi \colon V \times V \to \mathbb{R}$$

that $\forall x_1, x_2, y \in V$ and $\forall c \in \mathbb{R}$ satisfies the following axioms:

$$\Phi(x_1 + x_2, y) = \Phi(x_1, y) + \Phi(x_2, y) \tag{3.1a}$$

$$\Phi(cx, y) = c\Phi(x, y) \tag{3.1b}$$

$$\Phi(x, y) = \Phi(y, x) \tag{3.1c}$$

From the above, we can derive the following property

$$\Phi(y, x_1 + x_2) \overset{(3.1c)}{=} \Phi(x_1 + x_2, y) \overset{(3.1a)}{=} \Phi(x_1, y) + \Phi(x_2, y)\,, \tag{3.2}$$

which will be useful in subsequent developments. □

Symmetric bilinear form of the two vectors $x, y \in V$ can be seen as a generalised inner product. For instance, as a consequence of the above definition, the Euclidean

scalar product is defined as a specialised symmetric bilinear form which in addition to satisfying the above axioms also possesses the following property: for all non-zero vectors x in V $\Phi(x, x) > 0$.

Perhaps a more intuitive interpretation of symmetric bilinear forms is given by the notion of the *squared distance*, given in the following definition.

Definition 3.2 (Squared Distance)**.** Let V be a real vector space with a corresponding symmetric bilinear form Φ defined on it. The *square of the distance* between vectors x and y of V with respect to Φ is given by

$$\|x - y\|^2 = \Phi(x - y, x - y)\,.$$

In addition, a *squared norm* of vector x of V is defined as

$$\|x\|^2 = \Phi(x, x)\,. \qquad \square$$

Let integer n be the dimension of the vector space V. Also let x and y be any two vectors in V^n. If one chooses any basis $(a_i)_{1\leq i\leq n}$ of space V, the vectors x and y can be expressed via this basis as follows:

$$x = \sum_{i=1}^{n} x^i a_i \quad \text{and} \quad y = \sum_{i=1}^{n} y^i a_i\,.$$

Consequently, it follows from axiom (3.1b) of Definition 3.1, that the symmetric bilinear form $\Phi(x, y)$ can be evaluated as

$$\Phi(x, y) = \sum_{i=1}^{n}\sum_{j=1}^{n} x^i y^j \Phi(a_i, a_j)\,. \tag{3.3}$$

In particular, it follows from equation (3.3) that under the fixed basis $(a_i)_{1\leq i\leq n}$, the symmetric bilinear form is completely determined by the numbers $\Phi(a_i, a_j)$, called the *coefficients of the symmetric bilinear form* Φ with respect to the given basis. This leads to the following definition:

Definition 3.3 (Matrix of Symmetric Bilinear Form, Gram Matrix)**.** The square matrix

$$M(\Phi) = \big(\Phi(a_i, a_j)\big) \quad 1 \leq i, j \leq n$$

is called *the matrix of symmetric bilinear form* Φ with respect to the basis $(a_i)_{1\leq i\leq n}$. Furthermore, from axiom (3.1b) of Definition 3.1 it follows that the matrix $M(\Phi)$ is symmetric. It can also be seen, that the symmetric bilinear form Φ on the two vectors x and y could be expressed as

$$\Phi(x, y) = y^T M(\Phi) x\,. \qquad \square$$

The matrix of symmetric bilinear form $M(\Phi)$ with respect to the basis $(a_i)_{1\leq i\leq n}$ is often referred to as the *Gram matrix* (Gantmacher, 1959; Greub, 1967). In general, matrix $M(\Phi)$ completely defines the metric information for the corresponding vector space V.

Definition 3.4 (Taxonomy of Bilinear Forms)**.** A symmetric bilinear form Φ on a vector space V^n is said to be *non-degenerate*, if the rank of its matrix with respect to some basis of V is equal to n, and *degenerate* otherwise. In addition, Φ is called:

$$\begin{array}{ll} \textit{positive} & \text{if it is non-degenerate and } \forall x \in V \quad \Phi(x,x) \geq 0\,; \\ \textit{negative} & \text{if it is non-degenerate and } \forall x \in V \quad \Phi(x,x) \leq 0\,; \\ \textit{indefinite} & \text{if } \exists x, y \in V \text{ such that } \Phi(x,x) < 0 \text{ and } \Phi(y,y) > 0\ . \end{array}$$

□

The positive symmetric bilinear forms are usually called inner (scalar) products, and the vector spaces with inner products are widely known.

Example 3.1 (Lorentz Form)**.** Perhaps the most well known indefinite symmetric bilinear form is the Lorentz form (Greub, 1967) on $\mathbb{R}^4$ from the theory of relativity (Callahan, 2000). It is defined as

$$\Phi_L(x,y) = x^1y^1 + x^2y^2 + x^3y^3 - cx^4y^4\,, \tag{3.4}$$

where c is the speed of light. ▷

We will also need the notion of orthogonality in the vector space V, expressed via the symmetric bilinear forms:

Definition 3.5 (Orthogonality)**.** Vectors x and y of a vector space V with a symmetric bilinear form Φ on it are *orthogonal to each other with respect to* Φ, if $\Phi(x,y) = 0$. □

Finally, we present the following important property of the symmetric bilinear forms:

Proposition 3.1 (Relation of Symmetric Bilinear to Quadratic Form)**.** *The symmetric bilinear form can alternatively be expressed via squared distances in a vector space V (this relates the symmetric bilinear form to a notion of quadratic form (Gantmacher, 1959; Greub, 1967)):*

$$\Phi(x,y) = \tfrac{1}{2}\big(\Phi(x,x) + \Phi(y,y) - \Phi(x-y, x-y)\big)\,. \tag{3.5}$$

Proof. The proof uses the axioms of Definition 3.1 and the corollary (3.2).

$$\begin{aligned}\Phi(x-y,x-y) &\overset{(3.1a)}{=} \Phi(x,x-y)+\Phi(-y,x-y)\\ &\overset{(3.1b)}{=} \Phi(x,x-y)-\Phi(y,x-y)\\ &\overset{(3.2)}{=} \Phi(x,x)+\Phi(x,-y)-\Phi(y,x)-\Phi(y,-y)\\ &\overset{(3.1b)}{=} \Phi(x,x)-\Phi(x,y)-\Phi(y,x)+\Phi(y,y)\\ &\overset{(3.1c)}{=} \Phi(x,x)+\Phi(y,y)-2\Phi(x,y)\,.\end{aligned}$$

□

3.2.2 Pseudo-Euclidean Space

Definition 3.6 (Pseudo-Euclidean Space)**.** Let Φ be the non-degenerate symmetric bilinear form on a real vector space V of dimension n. A basis $(e_i)_{1\le i\le n}$ of V is called *orthonormal with respect to* Φ if the matrix of Φ with respect to it has the following canonical form

$$M(\Phi)=\begin{pmatrix} I_{n_+\times n_+} & 0\\ 0 & -I_{n_-\times n_-}\end{pmatrix}, \tag{3.6}$$

where $I_{n_+\times n_+}$ and $I_{n_-\times n_-}$ denote the identity matrices of dimensions n_+ and n_- respectively. The ordered pair of integers (n_+,n_-), where $n_++n_-=n$, is called the *signature of the form* Φ. The vector space V *together* with the form Φ is called a *pseudo-Euclidean (or Minkowski) vector space of signature* (n_+,n_-) and is denoted by $\mathbb{R}^{(n_+,n_-)}$. □

Given an orthonormal (w.r.t Φ) basis $(e_i)_{1\le i\le n}$ of V, the *inner product* between the two vectors $x,y\in V$ is measured as

$$\langle x,y\rangle=\Phi(x,y)=\sum_{i=1}^{n_+}x^iy^i-\sum_{j=n_++1}^{n}x^jy^j\ ,\ \text{where}\ x=\sum_{i=1}^{n}x^ie_i\ ,\ y=\sum_{i=1}^{n}y^ie_i\,.$$

The *square of the distance* between the two vectors in pseudo-Euclidean space is defined as

$$\|x-y\|^2=\Phi(x-y,x-y)=(x-y)^TM(\Phi)(x-y)\,,$$

where $M(\Phi)$ is the canonical matrix of the symmetric bilinear form given in equation (3.6).

The pseudo-Euclidean vector space $\mathbb{R}^{(n_+,n_-)}$ can be viewed as consisting of two *non-commensurable* Euclidean subspaces $\mathbb{R}^{n_+}$ and $\mathbb{R}^{n_-}$ of dimensions n_+ and n_-, respectively. If $n_-=0$, the pseudo-Euclidean space is Euclidean.

Example 3.2 (Minkowski Spacetime)**.** The Lorentz form Φ_L is given by equation (3.4) in Example 3.1. A pseudo-Euclidean space $\mathbb{R}^{(3,1)}$ corresponding to a pair $(\mathbb{R}^4,\Phi_L)$ is called *Minkowski spacetime* in special relativity theory (Pyenson, 1977; Rowe, 2001; Sexl and Urbantke, 2001). This Minkowski space consists of two non-commensurable subspaces $\mathbb{R}^3$ (space vectors) and $\mathbb{R}^1$ (time vectors). ▷

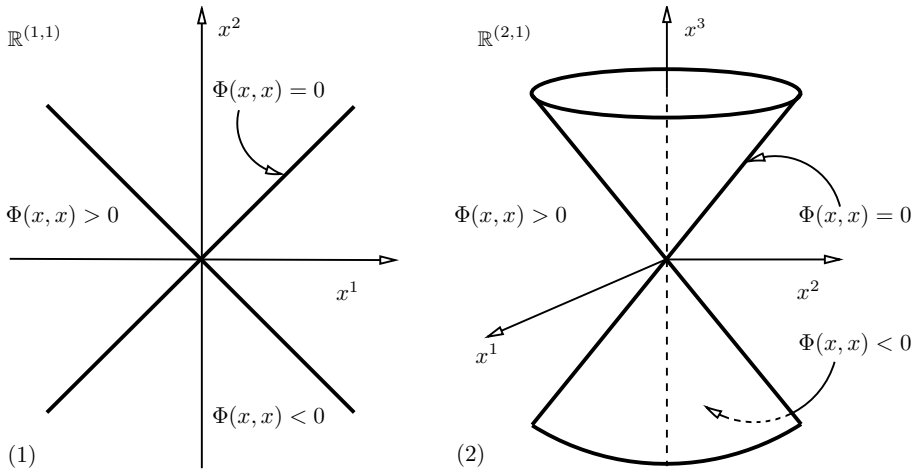


Figure 3.1: A visualisation of pseudo-Euclidean spaces $\mathbb{R}^{(1,1)}$ and $\mathbb{R}^{(2,1)}$ partitioned by the isotropic cones (Goldfarb, 1985).

Note that the square distances in pseudo-Euclidean space can be negative and, in particular, for a class of *indefinite* symmetric bilinear forms (such as the Lorentz form) there exist vectors x and y in V such that $\Phi(x,x) < 0$ and $\Phi(y,y) > 0$. The crucial difference between the Euclidean and pseudo-Euclidean vector spaces, however, is the existence in the latter of non-zero vectors x, called *isotropic*, which are orthogonal to themselves, i.e. such that $\Phi(x,x) = 0$. The set of all such isotropic vectors in V is called the *isotropic cone* of Φ and is amenable to geometrical interpretation. The inside of a cone consists of all vectors whose squared lengths are negative and the outside consists of the vectors with positive squared lengths, with the surface of the cone consisting of vectors of squared length zero (Goldfarb, 1985; Greub, 1967).

Example 3.3. Figure 3.1 shows the isotropic cones that partition pseudo-Euclidean spaces of signatures $(1,1)$ and $(2,1)$. A isotropic cone for a pseudo-Euclidean space of signature $(2,1)$ is shown on the right-hand side of Figure 3.1. The symmetric bilinear form corresponding to this space takes the form

$$\Phi(x,y) = x^1y^1 + x^2y^2 - x^3y^3$$

and the equation for the partition surface of an isotropic cone consisting of isotropic vectors is given by

$$\|x\|^2 = (x^1)^2 + (x^2)^2 - (x^3)^2 = 0\,. \qquad \triangleright$$

In the terminology of Greub (1967) adopted from special relativity theory, the isotropic vectors and isotropic cone correspond to *light vectors* and *light cone*, respectively. In special relativity the isotropic lines describe the trajectories of the photons in Minkowski spacetime (Callahan, 2000; Rowe, 2001).

Finally, it is desirable to have a method for relating the signature (n_+, n_-) of any (not necessarily non-degenerate) symmetric bilinear form Φ to the properties of the matrix $M(\Phi)$. According to the result proved by (Goldfarb, 1985, Theorem 3.12), for every vector space (V, Φ) of dimension n, there exists a basis of V with respect to which the matrix of Φ is

$$M(\Phi) = \begin{pmatrix} I_{n_+ \times n_+} & 0 & 0 \\ 0 & -I_{n_- \times n_-} & 0 \\ 0 & 0 & 0 \end{pmatrix}_{n \times n},$$

where the rank of $M(\Phi)$, called *the rank of* Φ, is $n_+ + n_-$, where $n \geq n_+ + n_-$.

Consequently, the following properties of Φ can be established via its signature:

Φ is *positive*	if and only if $n_+ = n$
Φ is *negative*	if and only if $n_- = n$
Φ is *indefinite*	if and only if $n_+ \geq 1$ and $n_- \geq 1$

3.3 From Metric to Pseudo-Euclidean Space: Isometric Embeddings

In the previous chapter, the phonological metric space was defined (Definition 2.4 on p. 59) as the set of objects together with the corresponding metric (or other) measure. With regard to speech, we defined the set of objects to be the set of structured objects that correspond to phonological templates $\mathbb{P}$. In addition, we provided several possible metrics $d_{\mathbb{P}}$ operating on that set.

Remark 3.1 (Naming Convention)**.** In the previous chapter we referred to the pair $(\mathbb{P}, d_{\mathbb{P}})$ as the metric space, while noting (see Remark 2.1) that the similarity measure $d_{\mathbb{P}}$ does not have to be a metric. In the following discussion, we will rectify this notational inconvenience by referring to the pair $(\mathbb{P}, d_{\mathbb{P}})$ as a *pseudo-metric* space. A pseudo-metric is a more general concept than a metric or semimetric, since on the one hand it is not constrained to obey the triangle inequality axiom and on the other allows zero distances between distinct objects (see Definition 2.3 in Section 2.2).

Furthermore, let P, where $|P| = k$, be a subset of a universe of all the phonological objects $\mathbb{P}$. Therefore, without loss of generality, when representing the pseudo-metric space, the similarity measure $d_{\mathbb{P}}$ can be replaced by the corresponding symmetric $k \times k$ matrix D_P of pair-wise proximities between the elements of a set P. In other words, the (phonological) pseudo-metric space is given by a pair (P, D_P). □

The natural question which arises next is how to make the transition to a vector space representation and what criteria should guide such a transition. In this section we

provide an answer to these questions. In general, given a pseudo-metric space (P, D_P), the goal is to reduce each of the original structural objects in P to a point in some abstract vector space where the decisions are to be made based on the metric information only. It is this information provided by the symmetric dissimilarity matrix between the objects of the set P which needs to be preserved by the embedding into the vector representation space. Hence, the embedding needs to be distance-preserving or *isometric*. An alternative, which we do not consider in this chapter, is to construct a vector space (not necessarily isometry-preserving) based on dissimilarities, where each original object in pseudo-metric space is represented in a new space by a $|P|$-dimensional dissimilarity-based feature vector, the elements of which specify the distances from the original object to the rest of the objects in the training set (Duin *et al.*, 2004; Pękalska, 2005; Pękalska *et al.*, 2004).

Most of the conventional approaches to dissimilarity-based multidimensional scaling for pattern representation and recognition (Borg and Groenen, 1997; Hérault *et al.*, 2002; Roth *et al.*, 2003) provide techniques for embedding the original metric spaces into the classical Euclidean vector space. It appears, however, that in many cases Euclidean space is not flexible enough to accommodate for an *isometric* embedding of the original problem and the "minimal" vector space in which such an isometry is always guaranteed to exist is pseudo-Euclidean (Goldfarb, 1985, Chapter 4). Because we are primarily interested in isometric transition from the proposed phonological metric space from the previous chapter to the vector space, in this chapter we consider a pseudo-Euclidean embedding, which is an efficient procedure for such a transition (Duin *et al.*, 2004; Goldfarb, 1984, 1985; Laub and Müller, 2004; Pękalska *et al.*, 2004). In Section 3.3.1 we describe one of the possible algorithms for an isometric pseudo-Euclidean space embedding, which we used in our work. In addition, we discuss dimensionality reduction which allows us, given the isometric embedding, to construct reduced vector representation of lower dimension by retaining the principal uncorrelated axes of the original sample. We provide a small, yet informative, example running throughout this section (Goldfarb, 1985, Example 4.1).

Example 3.4 (Impossibility of Euclidean Embedding)**.** Let the pseudo-metric space (P, D_P) be given by a set P of four objects $\{p_1, \ldots, p_4\}$ and the following dissimilarity matrix D_P

$$D_P = \begin{pmatrix} 0 & 1 & 2 & 1 \\ 1 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}_{4\times 4} . \tag{3.7}$$

Note that D_P has been generated by a *metric* because the triangle inequality is satisfied

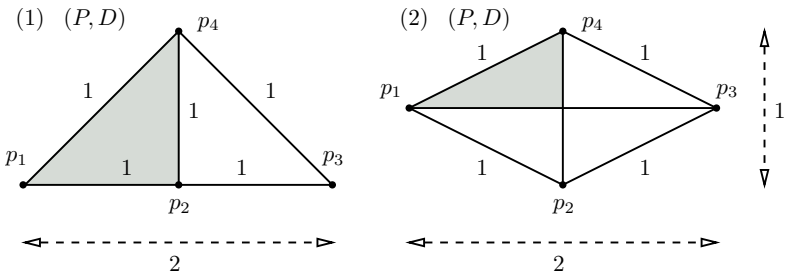


Figure 3.2: A diagrammatic representation of a four-dimensional metric space from Example 3.4 showing two possible configurations, both of which are impossible in Euclidean geometry.

for all pairs of points. This metric space is constructed by taking the first three points p_1, p_2 and p_3 on a straight line and then adding the point p_4 in such a way that it is of the equal distance to each of the first three points.

The diagrammatic representation of this metric space (having two potential configurations) is shown in Figure 3.2. It can be readily verified that the metric space represented by the dissimilarity matrix of equation (3.7) cannot be isometrically represented in a Euclidean space. We consider two possible configurations:

1. On one hand, p_4 must lie on the intersection of the two spheres of radius 1 with centers at p_1 and p_3 respectively and the point p_2 is the only point where these two spheres meet. On the other hand, p_4 must be at distance 1 from p_2.

2. Here the spheres of the radii 1 with centers at p_1 and p_3 do not intersect at any points, despite the fact that they should at both p_2 and p_4.

Alternatively, one can note that the theorem of Pythagoras is clearly violated by both configurations (see shaded triangles). The distance between p_1 and p_4 should be $\sqrt{2}$ for the first configuration and $\frac{\sqrt{5}}{2}$ for the second. ▷

3.3.1 Linear Embedding

Perhaps the most important result of (Goldfarb, 1985, Chapter 4) which we need is the existence of an isometric embedding in a pseudo-Euclidean space, stated in a following definition:

Definition 3.7 (Existence of Isometric Embedding)**.** Given a finite pseudo-metric space (P, D_P), $P = \{p_i\}_{i=1}^{k}$, there exists an *isometric embedding*

$$\alpha\colon (P, D_P) \to \mathbb{R}^{(n_+, n_-)},$$

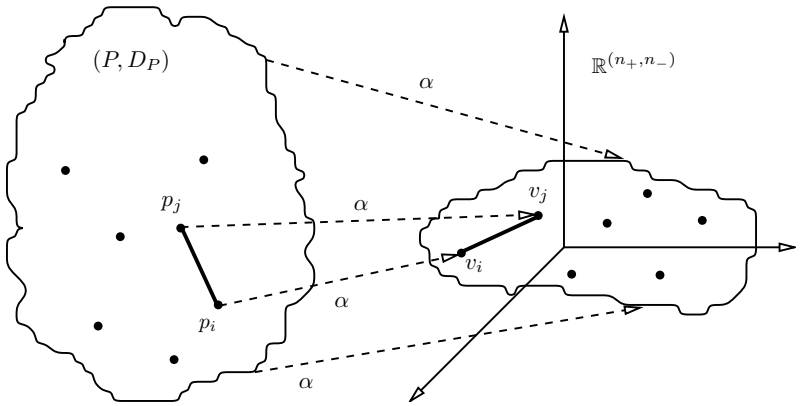


Figure 3.3: An isometric embedding $\alpha\colon (P, D_P) \to \mathbb{R}^{(n_+,n_-)}$.

which is called a *vector representation of* (P, D_P), such that for any other embedding of the same pseudo-metric space into a different space $\mathbb{R}^{(n'+,n'-)}$, the following condition

$$n'_+ \geq n_+ \quad \text{and} \quad n'_- \geq n_-$$

is true. In other words, the above property implies that vector representation α is the minimal mapping preserving isometry.

More formally, the notion of isometry can be introduced as follows: Let $v_i = \alpha(p_i)$, $1 \leq i \leq k$. Then for all objects p_i and p_j in a set P

$$D_P^2(p_i, p_j) = \|v_i - v_j\|^2 = \Phi(v_i - v_j, v_i - v_j) = \sum_{l=1}^{n_+} \left(v_i^l - v_j^l\right)^2 - \sum_{l=n_++1}^{n} \left(v_i^l - v_j^l\right)^2, \quad (3.8)$$

where Φ is the symmetric bilinear form of signature (n_+, n_-) corresponding to $\mathbb{R}^{(n_+,n_-)}$ (see Figure 3.3). □

3.3.1.1 Setting the Scene: Mean and Covariance in Pseudo-Euclidean Space

In order to introduce the notions of mean and covariance in pseudo-Euclidean space, we need to consider some system of vectors in that space. In line with Definition 3.7, assume that there exists an isometric embedding α' of the pseudo-metric space (P, D_P) of k objects onto the pseudo-Euclidean space (V', Φ). Let some $p_l \in P$, $1 \leq l \leq k$ map to the origin of the representation, i.e. $\alpha'(p_l) = 0$. Then, using equation (3.5), the inner product between any two vectors x_i and x_j in V' is given by

$$\Phi(x_i, x_j) \overset{(3.5)}{=} \tfrac{1}{2}\big(\Phi(x_i - 0, x_i - 0) + \Phi(x_j - 0, x_j - 0) + \Phi(x_i - x_j, x_i - x_j)\big) = \tfrac{1}{2}\big(D_P(p_i, p_l)^2 + D_P(p_j, p_l)^2 - D_P(p_i, p_j)^2\big). \quad (3.9)$$

The above coefficients of the symmetric bilinear form correspond to the matrix $M'(\Phi) = \Phi(x, y)$ of the symmetric bilinear form expressed solely on the basis of dissimilarities in

the original pseudo-metric space. This construction forms the basis of the initial isometric embedding algorithm (not treated here) proposed by (Goldfarb, 1985, Chapter 4). In particular, it is not difficult to see that the representation vectors x_i forming the basis of the space V' satisfy the isometric property (3.8) from Definition 3.7. Indeed, by using the result of Proposition 3.1, for all x_i and x_j in V'

$$\begin{aligned}\|x_i - x_j\|^2 &= \Phi(x_i, x_i) + \Phi(x_j, x_j) - 2\Phi(x_i, x_j) \\ &= \tfrac{1}{2}\big(D_P(p_i, p_l)^2 + D_P(p_i, p_l)^2 - D_P(p_i, p_i)^2\big) \\ &\quad + \tfrac{1}{2}\big(D_P(p_j, p_l)^2 + D_P(p_j, p_l)^2 - D_P(p_j, p_j)^2\big) \\ &\quad - \big(D_P(p_i, p_l)^2 + D_P(p_j, p_l)^2 - D_P(p_i, p_j)^2\big) = D_P(p_i, p_j)^2 \,.\end{aligned}$$

The *mean vector* for the vector representation $x_i = \alpha(p_i)$ is defined as

$$\overline{x} = \frac{1}{k}\sum_{i=1}^{k} x_i \,.$$

Note that the mean vector $\overline{x}$ of vector representation V may not necessarily have any direct physical interpretation in the original pseudo-metric space (P, D_P). In other words, it is not the case that there exists an object $\overline{p}$ in P such that $\alpha(\overline{p})$ is $\overline{x}$. Hence, during the derivation of the following identities, it is important to make sure that we do not rely on the existence of such an object in the original pseudo-metric space.

Using the axioms (3.1a), (3.1b) and the corollary (3.2) from the definition of symmetric bilinear form Φ, the norm of the mean with respect to Φ can be computed as

$$\begin{aligned}\Phi(\overline{x}, \overline{x}) &= \Phi\big(\tfrac{1}{k}(x_1 + x_2 + \ldots + x_k), \tfrac{1}{k}(x_1 + x_2 + \ldots + x_k)\big) \\ &= \frac{1}{k^2}\sum_{i=1}^{k}\sum_{j=1}^{k}\Phi(x_i, x_j) \\ &= \frac{1}{2k^2}\sum_{i=1}^{k}\sum_{j=1}^{k}\big(D_P(p_i, p_l)^2 + D_P(p_j, p_l)^2 - D_P(p_i, p_j)^2\big) \,.\end{aligned}$$

Then, by using the above identity, the squared distance between a mean vector $\overline{x}$ and

any vector x_i in V' can be expressed as follows

$$\begin{aligned}\|x_i - \overline{x}\|^2 &= \Phi(x_i - \overline{x}, x_i - \overline{x}) = \Phi(x_i, x_i) + \Phi(\overline{x}, \overline{x}) - 2\Phi(x_i, \overline{x}) \\ &= D_P(p_i, p_l)^2 + \frac{1}{k^2}\sum_{i'=1}^{k}\sum_{j=1}^{k}\Phi(x_{i'}, x_j) - \frac{2}{k}\sum_{j=1}^{k}\Phi(x_i, x_j) \\ &= D_P(p_i, p_l)^2 + \frac{1}{2k^2}\sum_{i'=1}^{k}\sum_{j=1}^{k}\big(D_P(p_{i'}, p_l)^2 + D_P(p_j, p_l)^2 - D_P(p_{i'}, p_j)^2\big) \\ &\quad - \frac{1}{k}\sum_{j=1}^{k}\big(D_P(p_i, p_l)^2 + D_P(p_j, p_l)^2 - D_P(p_i, p_j)^2\big) \\ &= \frac{1}{k}\sum_{j=1}^{k} D_P(p_i, p_j)^2 - \frac{1}{2k^2}\sum_{i'=1}^{k}\sum_{j=1}^{k} D_P(p_{i'}, p_j)^2 \,.\end{aligned} \tag{3.10}$$

An important consequence of the above equation (3.10) is that the calculation of the squared distance from any vector in the representation to the mean vector is independent of the chosen origin p_l.

Finally, we are ready to introduce the notion of the covariance in pseudo-Euclidean space.

Definition 3.8 (Covariance Matrix)**.** Let $\mathbb{R}^{(n_+,n_-)}$ be some pseudo-Euclidean space, the vectors of which are generated by the corresponding symmetric bilinear form Φ of signature (n_+, n_-) and let $\{\alpha(p_i)\}$ be the set of k vectors of dimension n representing some pseudo-metric space (P, D_P). Also, let vector $\overline{v}$ represent the mean vector of the vector representation α.

The set of all k vectors translated by the mean vector can be represented by the $n \times k$ matrix A whose ith column is $v_i - \overline{v}$. Then, the *(generalised) covariance matrix of the pseudo-metric space* (P, D_P) *with respect to vector representation* α is defined as the following $n \times n$ matrix

$$S_P(\alpha) = AA^T J = \Big(\sum_{i=1}^{k}(v_i - \overline{v})(v_i - \overline{v})^T\Big)\begin{pmatrix} I_{n_+ \times n_+} & 0 \\ 0 & -I_{n_- \times n_-} \end{pmatrix},$$

where J is the canonical matrix of symmetric bilinear form Φ in $\mathbb{R}^{(n_+,n_-)}$ from Definition 3.6. □

The above covariance matrix is more general than its Euclidean counterpart, because it consists of both negative and positive values. This fact is related to the non-commensurable properties of the two constituent subspaces of $\mathbb{R}^{(n_+,n_-)}$. Moreover, it has been shown (Goldfarb, 1985, Theorem 5.3) that all the characteristic values of $S_P(\alpha)$ are real numbers. This leads to the embedding algorithm, which is presented next.

3.3.1.2 Embedding Algorithm

In order to present the final result, we consider the original embedding α (see Definition 3.7). Unlike the embedding α', the assumption that representation V corresponding to α contains vectors which coincide with the origin is relaxed. Instead, we assume that *the origin of the space V coincides with the mean vector $\overline{v}$*, where the mean vector is not part of the representation set. Hence, given the pseudo-metric space (P, D_P), the coefficients $m_{i,j}$ of the matrix $M(\Phi)$ of the symmetric bilinear form are given by

$$\begin{aligned} m_{i,j} = \Phi(v_i, v_j) &\overset{(3.5)}{=} \tfrac{1}{2}\big(\Phi(v_i - \overline{v}, v_i - \overline{v}) + \Phi(v_j - \overline{v}, v_j - \overline{v}) - \Phi(v_i - v_j, v_i - v_j)\big) \\ &\overset{(3.10)}{=} \big(\frac{1}{k}\sum_{j'=1}^{k} D_P(p_i, p_{j'})^2 - \frac{1}{2k^2}\sum_{i'=1}^{k}\sum_{j'=1}^{k} D_P(p_{i'}, p_{j'})^2\big) \\ &\quad + \big(\frac{1}{k}\sum_{i'=1}^{k} D_P(p_{i'}, p_j)^2 - \frac{1}{2k^2}\sum_{i'=1}^{k}\sum_{j'=1}^{k} D_P(p_{i'}, p_{j'})^2\big) - D_P(p_i, p_j)^2 \,. \end{aligned}$$

By simplifying the above equation we obtain the following identity

$$\begin{aligned} m_{i,j} = \frac{1}{2}\Bigg[\frac{1}{k}\Big(\sum_{i'=1}^{k} D_P(p_{i'}, p_j)^2 + \sum_{j'=1}^{k} D_P(p_i, p_{j'})^2\Big) - \\ \frac{1}{k^2}\Big(\sum_{i'=1}^{k}\sum_{j'=1}^{k} D_P(p_{i'}, p_{j'})^2\Big) - D_P(p_i, p_j)^2\Bigg] \,. \end{aligned} \tag{3.11}$$

The above construction, based on the assumption that the origin of the resulting space coincides with the mean, has one important implication. In this case, it can be shown that the non-zero characteristic values of the $k \times k$ matrix $M(\Phi) = (m_{i,j})$ (where $m_{i,j}$ is given by equation (3.11)) of the symmetric bilinear form Φ of (P, D_P) coincide with those of the covariance matrix $S_P(\alpha)$ (see Definition 3.8) of (P, D_P) (Goldfarb, 1985, Theorem 6.3). Because the covariance matrix represents a reliable means of analysing the intrinsic dimensionality of the data, we adopt the following procedure, known as the (main) *embedding algorithm* (Goldfarb, 1984, 1985, 1986). It can be seen as a generalised version of conventional Principal Component Analysis (PCA), described by Duda *et al.* (2001).

Let (P, D_P) be the pseudo-metric space, where the set P consists of k objects. We want to construct an isometric vector representation of (P, D_P) in some pseudo-Euclidean space, the dimensions of which, at this point, are not known. Construction by the embedding α of a vector representation of (P, D_P) in a finite dimensional pseudo-Euclidean vector space is achieved by following the steps given below:

1. Compute the $k \times k$ matrix $M(\Phi)$ given by equation (3.11). By performing an eigen-decomposition of $M(\Phi)$, obtain the $k \times k$ matrix of the eigenvectors E and the

$k \times k$ diagonal matrix of the eigenvalues F, hence $M(\Phi) = EFE^T$. Several robust eigen-decomposition techniques exist. In this work, we use the QR technique, described in (Golub and Loan, 1983).

The eigenvalues in F correspond to the eigenvalues of the generalised covariance matrix. Therefore, the number of positive and negative eigenvalues in F, denoted by n_+ and n_- respectively, determine the dimension of the pseudo-Euclidean space for our isometric mapping. Hence, the dimension of this space is (n_+, n_-), where $n = n_+ + n_-$. The number of negligible eigenvalues (corresponding to noisy dimensions) of F is usually small, hence n is usually close to k.

2. Reorganise F into another $k \times k$ diagonal matrix C, which contains first the positive eigenvalues of $M(\Phi)$ in decreasing order, then the *magnitudes* of the negative eigenvalues in decreasing order, followed by zeros (if any are to be found among the eigenvalues of F). From E, construct the $k \times k$ matrix H of the eigenvectors of $M(\Phi)$ corresponding to the eigenvalues of $M(\Phi)$ in C. The matrix $M(\Phi)$ is now given by

$$M(\Phi) = HCH^T = HC^{\frac{1}{2}}\begin{pmatrix} J & \\ & 0 \end{pmatrix}C^{\frac{1}{2}}H^T = U\begin{pmatrix} J & \\ & 0 \end{pmatrix}U^T,$$

where $J_{n\times n}$ is a canonical matrix of Φ from (3.6).

3. Compute $k \times k$ matrix $U = HC^{\frac{1}{2}}$. The first $n_+ + n_-$ elements of the i-th row of U define the coordinates of $\alpha(p_i)$, of a vector representation $\alpha\colon (P, D_P) \to \mathbb{R}^{(n_+,n_-)}$ with respect to an orthonormal basis of $\mathbb{R}^{(n_+,n_-)}$.

An important consequence of the above algorithm is that any pseudo-metric space (P, D_P) can be isometrically represented in a classical Euclidean vector space *only if* all the numerically significant characteristic values of the corresponding generalised covariance matrix (or matrix of symmetric bilinear form) are positive.

Example 3.5 (Isometric Embedding)**.** The metric space, mentioned in Example 3.4, was defined on the four objects with the dissimilarity matrix D_P given by equation (3.7). We also mentioned that the isometric embedding could not be accommodated by Euclidean space. However, an isometric embedding into a pseudo-Euclidean space is possible and in this example we show how to construct such an embedding using the embedding algorithm described above.

The matrix of the symmetric bilinear form $M(\Phi)_{4\times 4}$, calculated using equation (3.11)

is

$$M(\Phi) = \begin{pmatrix} 0.94 & 0.06 & -1.06 & 0.06 \\ 0.06 & 0.19 & 0.06 & -0.31 \\ -1.06 & 0.06 & 0.94 & 0.06 \\ 0.06 & -0.31 & 0.06 & 0.19 \end{pmatrix}.$$

The corresponding 4×4 matrices F and E of eigenvalues and eigenvectors of $M(\Phi)$ are

$$F = \begin{pmatrix} -0.25 & 0 & 0 & 0 \\ 0 & 10^{-7} & 0 & 0 \\ 0 & 0 & 0.5 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}, \quad E = \begin{pmatrix} 0.5 & 0.5 & 10^{-7} & 0.70 \\ -0.5 & 0.5 & 0.70 & 10^{-7} \\ 0.5 & 0.5 & 10^{-8} & -0.70 \\ -0.5 & 0.5 & -0.70 & 10^{-8} \end{pmatrix}.$$

Since there are two non-negligible positive eigenvalues and one negative one, $n_+ = 2$ and $n_- = 1$. Hence, the metric space can be isometrically represented in $\mathbb{R}^{(2,1)}$. By computing the matrix U (not shown here) and discarding one noisy dimension, the 4×3 matrix V corresponding to the vector representation of (P, D_P) is given by

$$V = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{pmatrix} = \begin{pmatrix} \alpha(p_1) \\ \alpha(p_2) \\ \alpha(p_3) \\ \alpha(p_4) \end{pmatrix} = \begin{pmatrix} 1 & 10^{-7} & 0.25 \\ 10^{-7} & 0.5 & -0.25 \\ -1 & 10^{-8} & 0.25 \\ 10^{-8} & -0.5 & -0.25 \end{pmatrix}.$$

The mean vector $\overline{v}$ of the representation is equal to $(10^{-7}, 10^{-8}, 10^{-7})$ and, as expected, is very close to the origin. In addition, the matrix $D_V(v_i, v_j)$ of squared inter-distances between the vectors $\alpha(p_i)$ is given by

$$D_V^2 = \begin{pmatrix} 0 & 1 & 4 & 1 \\ 1 & 0 & 0.999999 & 1 \\ 4 & 0.999999 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

and hence, the isometry is preserved. The visualisation of V in $\mathbb{R}^{(2,1)}$ is shown in Figure 3.4. ▷

3.3.2 Dimensionality Reduction

Since the eigenvalues of $M(\Phi)$ correspond to the characteristic values of the generalised covariance matrix of the set $\{\alpha(p_i)\}$, the *reduced* vector representation (Goldfarb, 1985)

$$\beta \colon (P, D_P) \to \mathbb{R}^{(m_+, m_-)}, \quad m = m_+ + m_- < n$$

can be constructed from α by performing the following mapping

$$\gamma \colon \mathbb{R}^{(n_+, n_-)} \to \mathbb{R}^{(m_+, m_-)}, \tag{3.12}$$

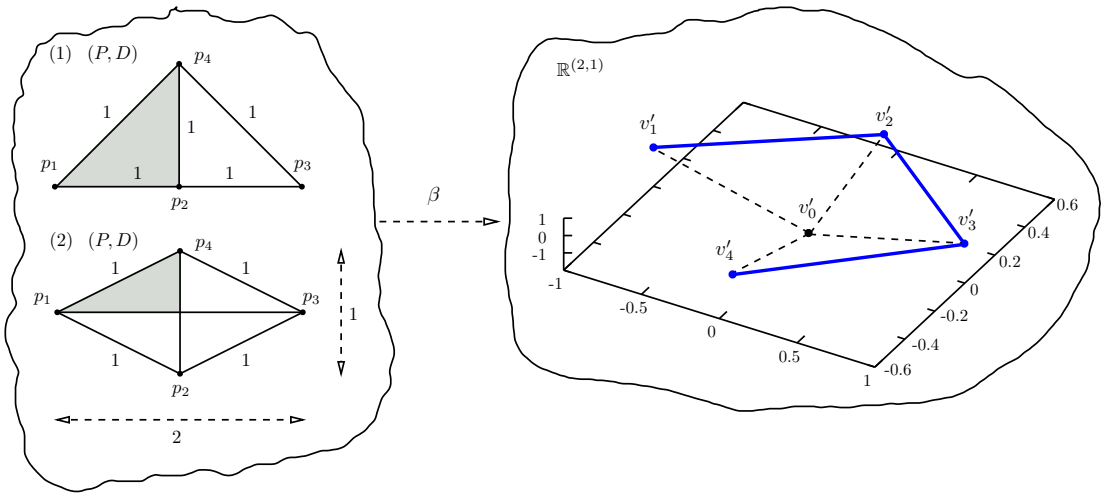


Figure 3.4: Minimal isometric embedding β of a metric space from Example 3.4 into a pseudo-Euclidean space $\mathbb{R}^{(2,1)}$. Vector v'_0 (not part of the representation) corresponds to the origin.

which is an orthogonal projection of the exact representation α on the subspace spanned by the corresponding principal axes of the covariance matrix.

This is accomplished by removing the axes corresponding to small magnitudes of the eigenvalues $|c_i|$ of C and retaining the eigenvalues corresponding to principal uncorrelated axes. If the removed eigenvalues are small, the resulting configuration $\beta = \gamma \circ \alpha$ possesses the same isometric properties as α since the orthogonal projection γ introduces only an insignificant perturbation to the original $k \times k$ dissimilarity matrix $(D_P(p_i, p_j))_{i,j}$. Pictorial representation of the above mappings β and γ is shown in Figure 3.5.

The vector representation constructed above by means of linear embedding α approximates the original inter distances between the objects exactly (Goldfarb, 1984, 1985). The intrinsic dimensionality of the data, however, might be much smaller and in practice, construction of the reduced vector representation using mapping β often removes the redundant and noisy dimensions from the original data. With the covariance matrix one has complete control over the dimensionality of the vector representation. Given a small positive threshold ϵ, one can remove all the axes whose respective eigenvalues c in the covariance matrix C are smaller (in magnitude) than ϵ.

The approach usually taken for dimensionality reduction is iterative, where one starts with the original dimension (n_+, n_-) of the embedding α and iteratively removes the dimensions corresponding to c_i which are considered to be correlated according to the criterion $|c_i| < \epsilon$, until no such dimensions are left (Goldfarb, 1985, 1986; Pękalska and Duin, 2002).

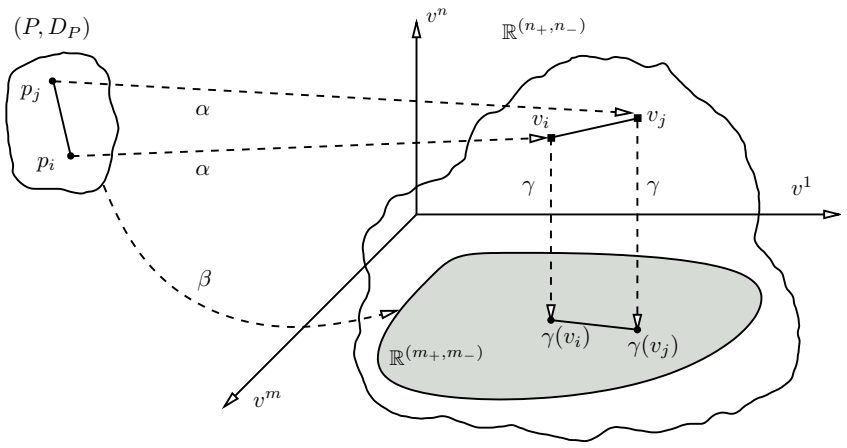


Figure 3.5: Reduced vector representation $\beta = \alpha \circ \gamma$ obtained by performing an orthogonal projection γ of the embedding α.

Given the reduced vector representation consisting of k vectors $v_i = \beta(p_i)$ in $\mathbb{R}^{(m_+,m_-)}$, the degree of how well the resulting configuration preserves the isometry can be expressed by a sum of squares error function (Goldfarb, 1986; Pękalska *et al.*, 2002):

$$E_\beta^D = (\sum_{i<j} \Phi_\beta(v_i, v_j)^2)^{-1} \sum_{i<j} (D_P(p_i, p_j)^2 - \Phi_\beta(v_i, v_j))^2 \,, \tag{3.13}$$

where Φ_β is the symmetric bilinear form of the resulting pseudo-Euclidean space $\mathbb{R}^{(m_+,m_-)}$. We refer to this measure as a *representation error*. An additional measure, which is useful when the iterative mode of dimensionality reduction is adopted, is the magnitude E_β^M of the eigenvalue removed at each step during the construction of the reduced representation. Obviously these two error measures are related: the bigger the magnitude of the removed eigenvalue, the bigger the increase in the representation error E_β^D, given by equation (3.13).

Example 3.6 (Metric "Line" in Pseudo-Euclidean Space)**.** This example (Goldfarb, 1985, Example 4.2) shows the construction of the reduced vector representation for an isometric embedding of a pseudo-metric space shown on the left-hand side of Figure 3.6, where a set P consists of eight objects and the dissimilarity matrix D_P is given by

$$D_P = \begin{pmatrix} 0 & 1 & 2 & 1 & 4 & 5 & 6 & 3 \\ 1 & 0 & 1 & 1 & 3 & 4 & 5 & 2 \\ 2 & 1 & 0 & 1 & 2 & 3 & 4 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 4 & 3 & 2 & 1 & 0 & 1 & 2 & 1 \\ 5 & 4 & 3 & 1 & 1 & 0 & 1 & 2 \\ 6 & 5 & 4 & 1 & 2 & 1 & 0 & 3 \\ 3 & 2 & 1 & 1 & 1 & 2 & 3 & 0 \end{pmatrix}_{8\times 8} . \tag{3.14}$$

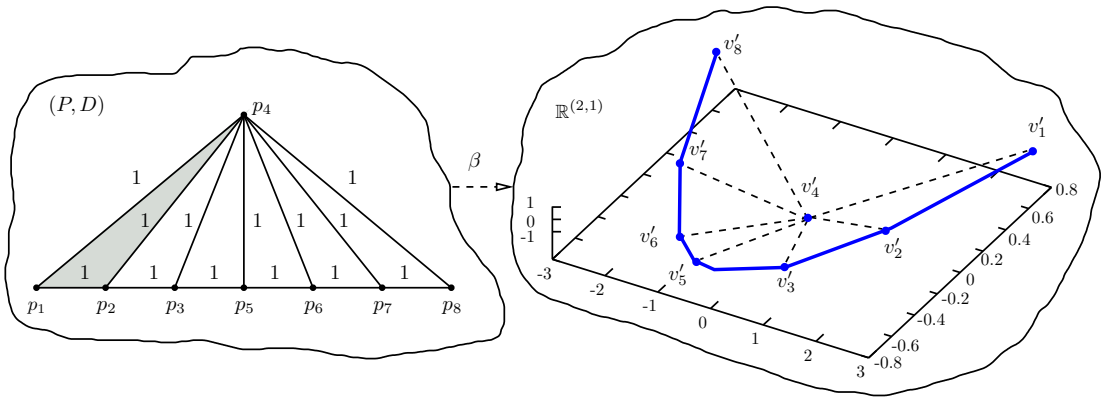


Figure 3.6: Reduced representation β of a metric space from Example 3.6 in a pseudo-Euclidean space $\mathbb{R}^{(2,1)}$. Vector v'_4 (part of the vector representation) corresponds to the origin.

The projection β is shown on the right-hand side of the figure (the representation was translated by the vector $\beta(p_4)$. Hence, v'_4 corresponds to the origin). It was obtained by first computing the isometric embedding α into an 8-dimensional pseudo-Euclidean space $\mathbb{R}^{(5,3)}$. The representation error E^D_β between (P, D_P) and α is close to 10^{-13}. Next, by setting the eigenvalue threshold ϵ to 10^{-6}, we obtained the projection β into a 3-dimensional pseudo-Euclidean space $\mathbb{R}^{(2,1)}$ with the representation error of 10^{-12}. An attempt to reduce this space by one dimension (by removing the eigenvalue $E^D_\beta = 1.78$ bigger than ϵ) results in the two-dimensional pseudo-Euclidean space $\mathbb{R}^{(1,1)}$ with a huge increase in the vector representation error (representation error in $\mathbb{R}^{(1,1)}$ is 0.013). Hence, we conclude that the intrinsic dimensionality of the pseudo-metric space (P, D_P) in question is 3.

The projection in Figure 3.6 has one interesting property: all points lie on a curve which could be called a "metric line" (Goldfarb, 1985). For every three consecutive points v'_i, v'_j and v'_k which belong to it,

$$\|v'_i - v'_j\| + \|v'_j - v'_k\| = \|v'_i - v'_k\|\,,$$

a property which in Euclidean space is only satisfied if the points belong to the same line. ▷

3.4 Projection of Unseen Objects

Let q be an object which did not participate in the construction of the reduced vector representation $\beta(p_i) \in \mathbb{R}^{(m_+,m_-)}$ of the training set $P = \{p_i\}_{1\leq i<k}$. In other words, q is in some set Q such that $P \cap Q = \emptyset$. The question that needs to be addressed next is

how to represent this new object in the constructed pseudo-Euclidean space $\mathbb{R}^{(m_+,m_-)}$. A procedure whereby q is added to the set P and the new configuration is obtained by re-embedding anew for each $q \in Q$ is obviously unacceptable for computational reasons.

A more feasible technique, which alleviates the above shortcoming, is to use the method of orthogonal projection. Expressed informally, this idea is as follows (Goldfarb, 1984, 1985): we assume that an isometric embedding α maps an object q onto *some* point $\alpha(q)$ in $\mathbb{R}^{(n_+,n_-)}$. Obviously, $\alpha(q)$ is not among the k points $\alpha(p_i)$ of a set comprising vector representation of a training set (P, D_P). Given this "phantom" $\alpha(q)$ in $\mathbb{R}^{(n_+,n_-)}$, it is possible to obtain its projection onto the reduced vector representation space $\mathbb{R}^{(m_+,m_-)}$ (a subspace of $\mathbb{R}^{(n_+,n_-)}$) by using only $m+1$ distance computations in (P, D_P).

In Section 3.3.1.1 it was mentioned that if we are given a set of k vectors v_i in a pseudo-metric space V corresponding to (P, D_P) and among these vectors there exists a vector p_0 such that its corresponding vector representation v_0 is zero, then the inner product between any two vectors v_i and v_j is given by equation (3.9) as:

$$\Phi(v_i, v_j) = \tfrac{1}{2}\big(D_P(p_i, p_0)^2 + D_P(p_j, p_0)^2 - D_P(p_i, p_j)^2\big)\,.$$

Based on the above observation, the basic idea is to somehow find the projection of q by computing the distance $D_P(q, p_0)$ between q and the object representing the origin, and m distances $D_P(q, p_i)$ between q and the m objects representing the basis of the reduced space $\mathbb{R}^{(m_+,m_-)}$. In Sections 3.4.1–3.4.2 we introduce two techniques for achieving this. The search for the m basis vectors of the reduced space $\mathbb{R}^{(m_+,m_-)}$, chosen among k vectors representing (P, D_P), is a non-trivial task described in Section 3.4.3.

3.4.1 Basic Metric Projection

We assume that the training procedure resulted in the construction of both complete and reduced representations α and β of the k vectors from the training set P in $\mathbb{R}^{(n_+,n_-)}$ and $\mathbb{R}^{(m_+,m_-)}$, respectively. *Basic* metric projection

$$\delta\colon (P, D_P) \to \mathbb{R}^{(m_+,m_-)}$$

of a new object q onto $\mathbb{R}^{(m_+,m_-)}$ proceeds as follows:

1. Among k vectors $v_i = \beta(p_i)$, choose a vector $v_0 = \beta(p_0)$ which would represent the origin. In this work, p_0 is chosen from P as the object whose average distance to the rest of the objects in the training set is minimal.

2. Perform a parallel translation

$$\tau\colon \mathbb{R}^{(m_+,m_-)} \to \mathbb{R}^{(m_+,m_-)}\,, \quad \tau(v_i) = v_i - v_0$$

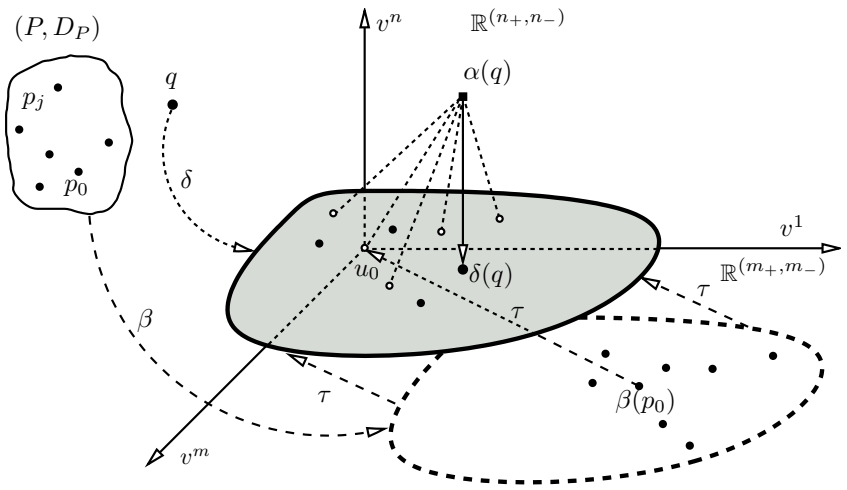


Figure 3.7: Basic metric projection δ of an unseen object q onto the reduced vector representation space $\mathbb{R}^{(m_+,m_-)}$. The chosen basis vectors are represented by the hollow circles.

of the k vectors v_i by v_0, hence $\tau(v_0) = 0$.

3. Among $k-1$ vectors $\tau(v_i)$ (the origin excluded) choose the basis

$$u_j = \tau(v_j)_{1 \le j \le m}$$

of $\mathbb{R}^{(m_+,m_-)}$ (see Section 3.4.3). In Definition 3.3 we mentioned that m vectors u_j of the basis completely define $\mathbb{R}^{(m_+,m_-)}$ via the corresponding $m \times m$ Gram matrix

$$G = \big(\Phi_\beta(u_i, u_j)\big) \quad 0 \le i, j \le m\,, \tag{3.15}$$

where Φ_β is a symmetric bilinear form of $\mathbb{R}^{(m_+,m_-)}$.

4. Orthogonal projection $\delta(q)$ of a new object q is defined by $m+1$ distances $D_P(q, p_0)$, $D_P(q, p_i)_{1 \le i \le m}$ as

$$\delta(q) = U_{m \times m} G^{-1}_{m \times m} b_{m \times 1}\,,$$

where columns of U are the coordinate columns of m vectors u_i and b is a vector whose ith coordinate is given by

$$b^i = \frac{1}{2}\big[D_P(q, p_0)^2 + D_P(p_i, p_0)^2 - D_P(q, p_i)^2\big]_{1 \le i \le m}\,. \tag{3.16}$$

Details of the derivation of the formulae in this step, closely related to the classical Gram orthogonal projection (Gantmacher, 1959, pp. 227–229), are given in (Goldfarb, 1985, Sections 3.3 and 7.2). Diagrammatic representation of the above algorithm is shown in Figure 3.7.

Since $B = UG^{-1}$ can be precomputed during the training stage, the only online computations involved are those of b and the product Bb in step (4). The rest of the computations in steps (1)-(3) are performed offline.

The Gram matrix G might be ill-conditioned, especially for large dimensions. Therefore, in this work we avoid calculating the direct inverse using the LU decomposition, but rather employ the Moore-Penrose pseudoinverse obtained using the Single Value Decomposition (SVD) technique (Albert, 1972; Laub, 2004).

3.4.2 Corrected Metric Projection

The reduced vector representation β gives an *approximation* of the original pseudo-metric space (P, D_P). Hence, the Gram matrix G for $\delta(p_i)$ in $\mathbb{R}^{(m_+,m_-)}$ in equation (3.15) differs from the exact Gram matrix for the $\alpha(p_i)$ in $\mathbb{R}^{(n_+,n_-)}$, while the calculations in (3.16) are based on the precise distances. In other words, the basic metric projection technique wrongly assumes that $\mathbb{R}^{(m_+,m_-)}$, given by the Gram matrix expressed via the symmetric bilinear form Φ_β, fully preserves the isometry.

In order to avoid the perturbation above, an alternative construction called *corrected* metric projection, referred to as δ_C,

$$\delta_C \colon (P, D_P) \to \mathbb{R}^{(m_+,m_-)}$$

is suggested in (Goldfarb, 1986). Informally, instead of choosing the m basis vectors of dimension m out of k translated vectors $\beta(p_i)$ in $\mathbb{R}^{(m_+,m_-)}$, a better idea is to choose the m basis vectors of dimension n out of k translated vectors $\alpha(p_i)$ in $\mathbb{R}^{(n_+,n_-)}$. Since α is an isometric mapping, the $m \times m$ Gram matrix for the newly chosen basis of $\mathbb{R}^{(n_+,n_-)}$ will employ the symmetric bilinear form Φ_α that preserves the original inter-object distances exactly.

More formally, this procedure is expressed as follows:

1. Choose an origin v_0 among the k vectors $v_i = \alpha(p_i)$ spanning $\mathbb{R}^{(n_+,n_-)}$.

2. Perform a parallel translation

$$\tau \colon \mathbb{R}^{(n_+,n_-)} \to \mathbb{R}^{(n_+,n_-)}, \quad \tau(v_i) = v_i - v_0$$

   of the k vectors v_i by v_0, hence $\tau(v_0) = 0$.

3. Among the k translated vectors choose m n-dimensional vectors

$$u_j = \tau(v_j)_{1 \le j \le m}$$

   in such a way that the signature of the $m \times m$ corresponding Gram matrix is (m_+, m_-).

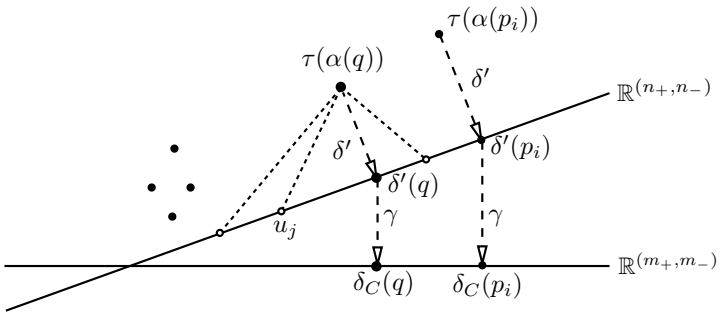


Figure 3.8: Corrected metric projection δ_C of an unseen object q onto the reduced vector representation space $\mathbb{R}^{(m_+,m_-)}$. The chosen basis vectors in $\mathbb{R}^{(n_+,n_-)}$ are represented by the hollow circles.

Here we would like to draw attention to the fact that unlike the corresponding step (2) of basic metric projection, m vectors u_j are not chosen to be the basis of $\mathbb{R}^{(n_+,n_-)}$ simply because $m < n$.

Furthermore, unlike the basic metric projection, the coefficients of the $m \times m$ Gram matrix G' are given here by the inner product Φ_α in $\mathbb{R}^{(n_+,n_-)}$.

4. Perform an orthogonal projection $\delta'(q)$ of a new object q onto $\mathbb{R}^{(n_+,n_-)}$ (note the difference with the corresponding step in the construction of basic metric projection) as

$$\delta'(q) = U_{n\times m} G^{-1}_{m\times m} b_{m\times 1}\,,$$

where columns of U are the coordinate columns of m vectors u_i and b is given by equation 3.16.

5. Finally, a corrected metric projection $\delta_C(q)$ of an object q is computed by applying the orthogonal projection

$$\gamma\colon \mathbb{R}^{(n_+,n_-)} \to \mathbb{R}^{(m_+,m_-)} \tag{3.17}$$

(defined in Section 3.3.2) to $\delta'(q)$ from the previous step. The above corrected metric projection procedure is depicted in Figure 3.8.

An additional important step, performed offline, is called the *correction* of the vector representation in $\mathbb{R}^{(m_+,m_-)}$. This step is not present in the construction of the basic metric projection. The idea is to apply the above steps (3) and (4) to *all* the vectors in the training set P. In other words, we apply the corrected metric projection δ_C to all of the k objects $p_i \in P$. This is essentially an orthogonal projection of all the k vectors $\tau(\alpha(p_i))$ onto the subspace of $\mathbb{R}^{(n_+,n_-)}$ spanned by m basis vectors u_j and therefore isomorphic to $\mathbb{R}^{(m_+,m_-)}$.

The following remark highlights a very important property of all the constructions (embedding, reduced representation and two metric projections) presented up to this point. This property affects the construction of *decision surfaces* (sometimes known as separating hyperplanes (Greub, 1967)) in pseudo-Euclidean spaces (Goldfarb, 1985, Section 7.3):

Remark 3.2 (Decision Surfaces)**.** Let $\mathbb{R}^{(m_+,m_-)}$ be the the pseudo-Euclidean vector space in which the problem is represented by the vectors $v_1, \ldots, v_m$. Select *any* pair of vectors v_i and v_j. Let U denote the hyperplane in $\mathbb{R}^{(m_+,m_-)}$ which is orthogonal to the segment $s_{i,j}$ connecting v_i and v_j. Also let $\hat{v}$ be the the vector in $\mathbb{R}^{(m_+,m_-)}$ normal to the plane U. Since the normal vector $\hat{v}$ is parallel to $s_{i,j}$, $\|\hat{v}\| \geq 0$.

In particular, a positive norm implies that the squared distances (w.r.t. corresponding symmetric bilinear form) between all the vectors representing the training data, as well as the projections of unseen objects, are positive. Furthermore, in most of the cases, classical decision surface algorithms for Euclidean spaces can safely be used. In some cases, however, the decision algorithms make implicit assumptions about the geometry of the underlying space. Linear support vector classifiers, for instance, require the form of the dissimilarity matrix between the objects to be positive definite (Schölkopf and Smola, 2001). In such cases the algorithms have to be generalised to operate in pseudo-Euclidean spaces.

An alternative is to treat the resulting pseudo-Euclidean space as a regular Euclidean vector space with a positive definite matrix of symmetric bilinear form. If such a simplifying assumption is made any Euclidean-space classifier can be used (we treat this case in more detail in Section 3.5.1.3). □

3.4.3 Selecting the Basis for Metric Projections

For both metric projection methods, δ and δ_C, the m basis vectors spanning $\mathbb{R}^{(m_+,m_-)}$ are chosen in such a way as to minimise the *average projection error* between the projection of the entire training set (obtained with δ or δ_C) and the original vector representation of a pseudo-metric space (P, D_P) obtained by the mappings α or β.

Definition 3.9 (Projection Error)**.** Let $\hat{v} \in \mathbb{R}^{(m_+,m_-)}$ be the projection of a vector $v \in \mathbb{R}^{(n_+,n_-)}$ onto the subspace $\mathbb{R}^{(m_+,m_-)}$ of $\mathbb{R}^{(n_+,n_-)}$. The projection error between two vectors v and $\hat{v}$ is defined as the squared distance between them (Goldfarb, 1985), i.e. as

$$\Phi(v - \hat{v}, v - \hat{v}) = \|v - \hat{v}\|^2 = D_P(p, p_0)^2 - bG^{-1}b^T\,, \tag{3.18}$$

where p_0 is the selected origin, b is defined in (3.16) and G is the Gram matrix w.r.t the basis. □

In what follows, we assume that the vector representation in $\mathbb{R}^{(m_+,m_-)}$ is given by the $k \times m$ matrix U, whose rows are the vectors representing the objects from the original pseudometric space. We next introduce two approaches to the selection of the optimal basis. The first approach to basis selection, which we refer to as *regular*, was suggested in (Goldfarb, 1986; Pękalska, 2005; Pękalska *et al.*, 2002). We also present an alternative construction, called *class-based* selection, designed by us to address some of the potential problems with the first approach.

3.4.3.1 Regular Basis Selection

The search for the optimal subset U_r of m basis vectors among k vectors comprising the representation U is an iterative process. Assume the algorithm is currently at step l. At the previous step, $l-1$ objects have been selected as the basis U_r^{l-1} forming a subspace of $\mathbb{R}^{(m_+,m_-)}$. The next candidate to be added to the basis is chosen from the set $U \setminus U_r^{l-1}$ as the one minimising the average error between the resulting projection of all the vectors in U onto $\mathbb{R}^{(l_+,l_-)}$ (given by equation (3.18) where both G and b are calculated w.r.t candidate set U_r^l) and the original vector representation.

3.4.3.2 Class-based Basis Selection

Note that in the regular approach, described above, at each step the class label of the candidate vector is ignored. This often results in uneven representation of classes within the basis of the reduced space. In order to verify how taking into account the class labels affects the classification performance, we implemented a novel alternative basis selection approach. This approach selects the basis of the reduced space in such a way that the projection error is kept to minimum, while making sure that the vectors comprising the basis are well-balanced in terms of class representation.

Informally, this method operates as follows: Assume that the set U is subdivided into N classes, i.e. $U = C_1 \cup C_2 \ldots \cup C_N$. The algorithm proceeds in exactly the same fashion as its regular version but with one important distinction. At iteration l, in addition to keeping track of the number of vectors $l-1$ currently selected as the basis, we also keep the identity $i-1$ of the class C_{i-1} to which the last selected vector belongs. The next lst vector is chosen not from the entire set $U \setminus U_r^{l-1}$, but rather from the next class $C_i \setminus C_i^{l-1}$, where C_i^{l-1} is the set of vectors from class C_i which already reside in the basis at iteration l.

3.5 Experiments and Discussion

In this section we present the experimental results of a phone classification task on the data represented in the pseudo-Euclidean domain. The original symbolic data consists of phonological feature templates derived from the TIMIT database of read speech (Garofolo, 1988; Garofolo *et al.*, 1993). Phonological feature templates, which make up a phonological metric space, were described in detail in Chapter 2. The details of the TIMIT corpus, as well as the procedures for deriving the phonological templates from speech are given in Section 2.6.

Briefly, our phonological metric space consists of two sets — one for training and one for testing, with 176,031 phone tokens from 39 classes. In Section 2.6, we described the classification results obtained on this task in the symbolic metric space, which was defined over a set of phonological templates. In that chapter, various symbolic metric algorithms were evaluated on several quantised datasets (corresponding to several pseudo-metric spaces, one for each quantisation level) obtained from TIMIT. The pseudo-metric space (P, D_P) we have chosen in this chapter as the basis for our experiments, corresponds to the phonological metric space of Section 2.6 for which the best classification results were obtained. The details of (P, D_P) are as follows:

- The set P corresponds to the set of phonological templates derived from the TIMIT data using a symbolic quantisation level of 10. The symbolic corpus corresponding to this quantisation level consists of 124,962 templates in the training set and 42,540 templates in the test set.

- The similarity function D_P, operating on the phonological templates from the set P, is the weighted Levenshtein distance.

- The clustering technique is k-medians, employing phonological set median (rather than generalised median) and duration-based initialisation.

Since the above dataset is large, for the experiments described below we have adopted the following strategy: in order to get a better idea of the performance of the classification algorithms in the pseudo-Euclidean space constructed from (P, D_P), we first focus on a smaller 3-class task. The classes used in this task belong to three different phonological categories and are *a priori* known to be reasonably well separable, according to the linguistic evidence (Ladefoged, 2001). This allowed us to test a wider range of the classifiers, the results of which are otherwise difficult to interpret on a larger 39-class task. Experiments on a 3-class problem are described in Section 3.5.1. In order to compare the performance of the classifiers in the original symbolic space

(Section 2.6) and the pseudo-Euclidean space, in Section 3.5.2 we describe the results of the experiments on a full 39-class task.

3.5.1 Three-class Problem

The first set of experiments focuses on classification of three classes of phonemes from three different phonological categories. The three classes under investigation consist of one vowel [aw] (low back round) and two consonants [b] (voiced bilabial stop) and [z] (voiced alveolar fricative). The original training set for these three classes consists of 6,629 unique symbolic phonological templates. The entire test set for the three classes of phones, consisting of 2,423 unique phonological templates, was used in this experiment.

Since the matrix (D_P) of pseudo-metric space interdistances for such a set is rather large and can cause numerical stability problems for matrix decomposition algorithms, we reduced the dimensionality of the training set to 100 phonological templates per class, using the clustering procedure in the original symbolic space (P, D_P). The algorithmic setup for the symbolic clustering is described above.

3.5.1.1 Visualisation

In order to obtain the visualisation of the three-class problem, we first constructed an isometric embedding α (see Section 3.3.1) for the pseudo-metric space (P, D_P) consisting of 300 objects. The resulting isometric pseudo-Euclidean space is

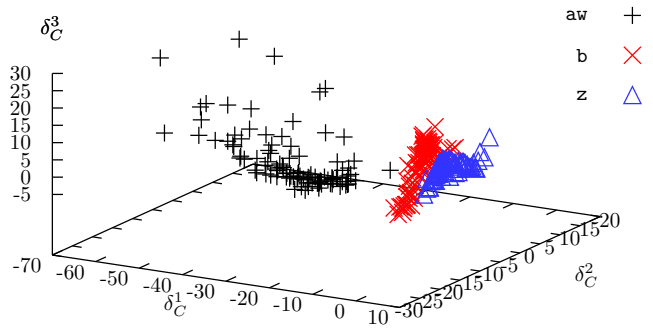
$$\mathbb{R}^{(143,156)}, \quad \text{where} \quad n = 299\,. \tag{3.19}$$

Obviously, this space is highly pseudo-Euclidean, which means that the three-class problem can not be isometrically represented by any conventional Euclidean vector space. Next, we constructed a corrected metric projection δ_C (see Section 3.4.2) of the entire training (300 vectors) and test sets (2,423 vectors) onto a pseudo-Euclidean space of the same dimension. This step performs the calibration of the training and test sets. The resulting projections, visualised by the three principal axes, are shown in Figure 3.9 (p. 107).

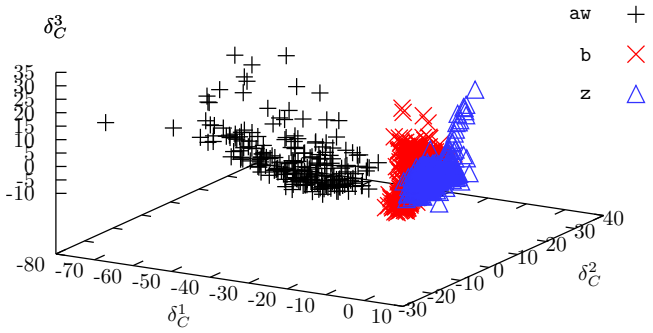
The visualisation of the three principal axes of the corrected metric projection appears to confirm the initial hypothesis that the chosen metric is adequate for discriminating between the classes in question and that some reasonably simple linear decision surfaces can be constructed in the resulting vector representation space.

3.5.1.2 Dimensionality Reduction

The vector representation constructed above by means of linear embedding α approximates the original interdistances between the patterns exactly (Goldfarb, 1985). Recall



(a) Training set.



(b) Test set.

Figure 3.9: Corrected metric projection δ_C of the training (3.9a) and test (3.9b) sets for the three-class problem.

from the discussion in Section 3.3.2, that the intrinsic dimensionality of the data might be much smaller and in practice, construction of the reduced vector representation using mapping β often removes the redundant and noisy dimensions from the original representation (Goldfarb, 1985, 1986; Pękalska *et al.*, 2002).

Let m be the dimension of the reduced vector representation β. The degree of how well the resulting configuration preserves the isometry can be expressed by three different measures:

- The sum of squares error function E_β^D between the original isometric representation and the reduced one, which is given by equation (3.13) from Section 3.3.2.

- The magnitude of the eigenvalue E_β^M removed at each step of the construction of the reduced representation, described in Section 3.3.2.

- The average class separability measure $\overline{D}_\beta$, which is calculated as follows: Let P be the training set divided into N classes P_l, $P = P_1 \cup P_2 \ldots \cup P_N$. The *squared*

average within-class distance for a class P_l is given by

$$\overline{D}_W(P_l)^2 = \frac{2}{|P_l|(|P_l|-1)} \sum_{i=1}^{|P_l|} \sum_{j=i+1}^{|P_l|} \|\beta(p_{l_i}) - \beta(p_{l_j})\|^2 .$$

The *squared average between-class distance* for a pair of classes P_k and P_l is given by

$$\overline{D}_B(P_k, P_l)^2 = \frac{1}{|P_k||P_l|} \sum_{i=1}^{|P_k|} \sum_{j=1}^{|P_l|} \|\beta(p_{k_i}) - \beta(p_{l_j})\|^2 .$$

The *average class separability* is then defined as

$$\overline{D}_\beta = \frac{1}{(N-1)(N-2)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \frac{\sqrt{|\overline{D}_B(P_i, P_j)^2|}}{\sqrt{|\overline{D}_W(P_i)^2|} + \sqrt{|\overline{D}_W(P_j)^2|}} .$$

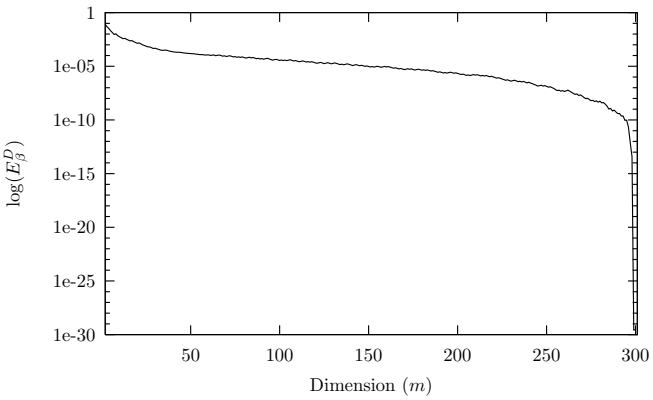
Given the original isometric representation α of 300 training objects, we conducted dimensionality reduction experiments in order to study the behaviour of pseudo-Euclidean spaces of smaller dimensionality. The representation error measures E_β^D, E_β^M and $\overline{D}_\beta$ (the first two measures are plotted using log scale) are shown in Figure 3.10 (p. 109) against the dimension m of the corresponding reduced pseudo-Euclidean space. The upper bound on the reduced dimension was chosen to be 150.

We also performed an analysis of the eigenvalues removed during the reduction of dimensionality and established that for all dimensions m bigger than 9, the reduced space is strictly pseudo-Euclidean.

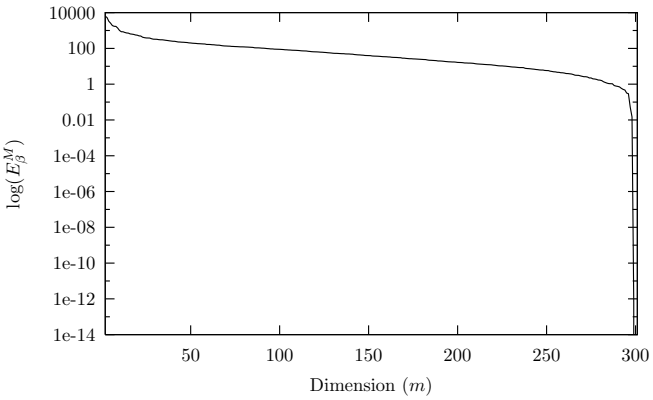
3.5.1.3 Classification

In order to evaluate the performance of various classifiers constructed in pseudo-Euclidean space, we conducted several classification experiments on a three-class task described above. The baseline setup was chosen to correspond to the symbolic methods which perform the best on the three-class problem in the original phonological metric space (P, D_P). The best performing symbolic classifier (which performs the best on both the small and full class problems) is the k-NN AESA search (see Section 2.6) based on the score of the top candidate (in terms of the smallest distance to the test token) in the k-best list, i.e. $k = 1$. This was found to outperform all the majority voting schemes. The classification error obtained with 1-NN AESA (for the setup described above, with the training set of 300 templates and full test set of 2,423 templates) is 0.9%.

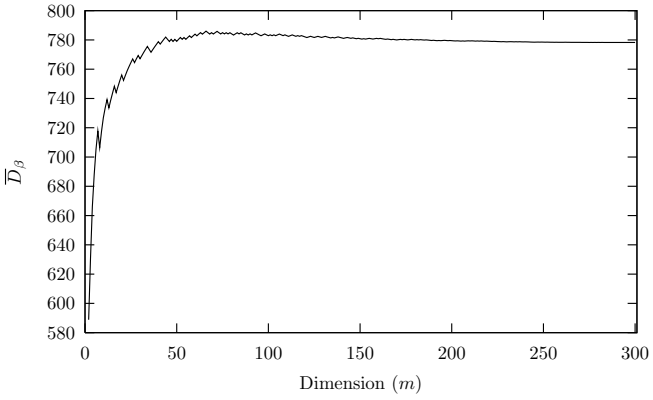
In Remark 3.2 in Section 3.4.2, an important property of the pseudo-Euclidean vector space representation was stated. According to this property, the squared lengths of the segments connecting all the vectors representing the training and test objects are non-negative. This allows one to use extensions of classical decision algorithms



(a) Log of the representation error E_β^D.



(b) Log of the magnitude of removed eigenvalue E_β^M.



(c) Average class separability $\overline{D}_\beta$.

Figure 3.10: Dimensionality reduction for the three class problem: Error measures as functions of the reduced dimension m of $\mathbb{R}^{(m_+,m_-)}$.

available in Euclidean spaces. In addition, some of the classification rules, like k-NN, are universal and apply to symbolic spaces too. Following are the classifiers we used in the experiments:

kNN Perhaps the simplest classifier one can use in the pseudo-Euclidean space is based on the k Nearest Neighbour (k-NN). Goldfarb (1985) showed that the k Nearest Neighbour rule is suitable for any vector space (including the pseudo-Euclidean) provided that the similarity between the vectors is expressed with the help of the inner product (symmetric bilinear form) corresponding to that space. We extended the efficient k-Approximating and Eliminating Search Algorithm (k-NN AESA), described in Section 2.5.4, to operate in pseudo-Euclidean spaces by changing the specification of the algorithm to operate on squared distances in pseudo-Euclidean space, rather than on Euclidean distances. The k-NN AESA search in pseudo-Euclidean spaces is hereafter referred to as *kNN*.

LDS Given the reduced vector representation and noticing that the metric is nearly linearly separating the classes in question, we proceed by constructing a simple linear decision surface by using a feed-forward neural network without any hidden units, hereafter referred to as a *LDS*. The feed-forward neural network has a simple architecture. It performs 1-of-3 classification. The number of input (activation) units is equal to the dimension m of the feature vectors in $\mathbb{R}^{(m_+,m_-)}$. There are 3 output (target) units. Each input unit is mapped to all three output units. The NICO artificial neural network toolkit was used as implementation (Ström, 1996).

SVM An additional classifier we use is the Support Vector Machine (Cortes and Vapnik, 1995; Schölkopf and Smola, 2001; Vapnik, 1998). The basic training principle behind the SVM is finding the optimal linear hyperplane such that the expected classification error for unseen test samples is minimised. According to the structural risk minimisation inductive principle (Burges, 1998; Vapnik, 1998), a function that classifies the training data accurately and which belongs to a set of functions with the lowest VC dimension (Burges, 1998; Cortes and Vapnik, 1995) will generalise best regardless of the dimensionality of the input space. Based on this principle, a linear SVM uses a systematic approach to find a linear function with the lowest VC dimension. For linearly non-separable data, SVMs can (nonlinearly) map the input to a high dimensional feature space where a linear hyperplane can be found.

Given a labelled set of N training samples (x_i, y_i), where each vector x_i is assigned a class membership criteria $y_i \in \{+1, -1\}$, the SVM classifier finds the optimal

hyperplane that correctly separates (classifies) the largest fraction of vectors while maximising the distance of either class from the hyperplane (the margin). Maximising the margin distance is equivalent to minimising the VC dimension in constructing an optimal hyperplane. Computing the best hyperplane is posed as a constrained optimisation problem and solved using quadratic programming techniques. The discriminant hyperplane is defined by the level set of

$$f(x) = \sum_{i=1}^{N} y_i \alpha_i k(x, x_i) + b\,,$$

where $k(\cdot, \cdot)$ is a kernel function and the sign of $f(x)$ determines the membership of x. Constructing an optimal hyperplane is equivalent to finding all the nonzero α_i. Any vector x_i that corresponds to a nonzero α_i is a support vector (SV) of the optimal hyperplane.

In his work, Graepel (1999) made the first attempt to use support vector classifiers on a non-Euclidean data[1]. He found out that when the vector space in question is pseudo-Euclidean, the constrained optimisation can not be performed meaningfully if the $N \times N$ matrix of the kernel values

$$K = \big(k(x_i, x_j)\big), \quad 1 \le i, j \le N$$

corresponding to the training data is not positive definite. In the simplest scenario, this can be demonstrated on a linear SVM classifier. In this case, the kernel function is specified by the squared pseudo-Euclidean distances between the points in the training set, giving rise to K which corresponds to the pseudo-Euclidean dissimilarity matrix which is often not positive definite (in our task, since the modelling space has a very strong negative component according to equation (3.19), this matrix is indefinite). As the result, the optimisation problem is ill posed: it is still quadratic, but not convex.

Two workarounds have been proposed by Graepel *et al.* (1999) and Pekalska *et al.* (2005; 2002). Both of these methods essentially disregard the geometry of pseudo-Euclidean space and either perform classification in $\mathbb{R}^{(m_+, m_-)}$ assuming it is a Euclidean space $\mathbb{R}^{m_+ + m_-}$ or perform all the projections on the real part $\mathbb{R}^{m_+}$ of the space only. In other words, all the inner product computations performed by the kernel functions are computed in a Euclidean space. This technique performed well in the reported experiments.

In our experiments, we follow the approach taken by Pękalska *et al.* (2002) and perform all the computations in $\mathbb{R}^{(m_+, m_-)}$ using the Euclidean, rather than pseudo-

[1] The data for which symmetric bilinear form is not positive definite (Section 3.2.1).

Euclidean, inner products. We used the SVMtorch toolkit as the implementation (Collobert and Bengio, 2000, 2001). Hereafter, we refer to this classifier as *SVM*.

First, we constructed isometric embedding α (Section 3.3.1) of a full training set of 300 patterns. Next we constructed reduced representations of α with the dimension of the biggest reduced representation corresponding to 150. The motivation behind chosing the latter value is to use less than 50% of the original training data.

Next, four classification experiments were conducted with each of the classifiers described above and the results were compared to the best classification error of 0.9% obtained in the pseudo-metric space with 1-NN AESA. For each of the classifiers and each of 150 dimensions m, characterising the reduced pseudo-Euclidean space, we performed the experiments with the following four possible configurations:

- Two different reduced vector representations are constructed using regular (or basic) δ (denoted by subscript R) and corrected δ_C (denoted by subscript C) metric projections (Section 3.4).

- During the construction of each of the reduced vector representations, two different approaches to the basis selection of the reduced vector space $\mathbb{R}^{(m_+,m_-)}$ were employed: regular (denoted by superscript R) and novel class-based (denoted by superscript C) basis selection techniques (Section 3.4.3).

Best classification results (chosen from reduced dimensions of $m \leq 150$) for each of the resulting configurations, along with the corresponding dimension m for which the results were obtained, are shown in Table 3.1.

For *kNN* classifiers, simple nearest neighbour 1-NN search based on the score of the top candidate (in terms of the smallest distance to the test pattern) in the k-best list outperformed the majority voting schemes. Interestingly enough, the same situation was encountered for the symbolic version of k-NN AESA.

We first compare the performance of the neural networks to the performance of 1-NN search. As can be seen from Table 3.1, feed-forward neural networks (*LDS*) consistently outperform k-NN search in both pseudo-metric and pseudo-Euclidean spaces. In addition, the use of class labels for the construction of the reduced vector representation during the training stage, improves the performance of all the *LDS* and *kNN* models used in this experiment. The 1-NN classifier in the *reduced* pseudo-Euclidean space does not seem to handle perturbations introduced by the dimensionality reduction as well as the neural networks. Its error rate, however, comes close to its symbolic counterpart by *only* using around 17% (50 out of 300 patterns) of the original training data. The best

Classifier	Projection	Basis Selection	Notation	Dimension	Error (%)
kNN AESA	Regular	Regular	kNN_R^R	85	2.8
	Corrected	Regular	kNN_C^R	73	2.4
	Regular	Class-based	kNN_R^C	63	1.1
	Corrected	Class-based	kNN_C^C	50	**1.0**
LDS	Regular	Regular	LDS_R^R	39	0.7
	Corrected	Regular	LDS_C^R	144	0.6
	Regular	Class-based	LDS_R^C	33	0.6
	Corrected	Class-based	LDS_C^C	130	**0.5**
SVM	Regular	Regular	SVM_R^R	48	**0.3**
	Corrected	Regular	SVM_C^R	54	0.5
	Regular	Class-based	SVM_R^C	92	0.4
	Corrected	Class-based	SVM_C^C	87	0.5

Table 3.1: Various types of classifiers used in the experiments on different types of pseudo-Euclidean representations and the corresponding best error rates.

performing neural network model achieved 0.5% error, which is an improvement over the best pseudo-Euclidean 1-NN result of 1.0% and symbolic 1-NN result of 0.9%.

We next turn to analysing the performance of the linear support vector classifiers (SVM), noticing that they too consistently outperform 1-NN search in both pseudo-Euclidean and original pseudometric spaces. These findings are in line with the previously reported observations by Graepel *et al.* (1999); Pękalska and Duin (2002); Pękalska *et al.* (2002). As can be seen from Table 3.1, all the four configurations of the linear support classifiers perform as well or better than their corresponding neural network counterparts, with the best LDS errors corresponding to the worst SVM ones. Interestingly enough, with the linear support classifiers, regular basis selection algorithm performs better than its class-based counterpart, on both spaces obtained by basic and corrected metric projections. This could be explained by the fact that the ratio between the number of objects per class and the overall number of classes for this task is reasonably high. Hence, the classes are represented reasonably fairly in the chosen basis. Fair representation of classes becomes more of an issue when the task is sparse, i.e. when there are too many classes and too few objects to represent them. Therefore, we expect the class-based technique for basis selection to perform better in sparse environments.

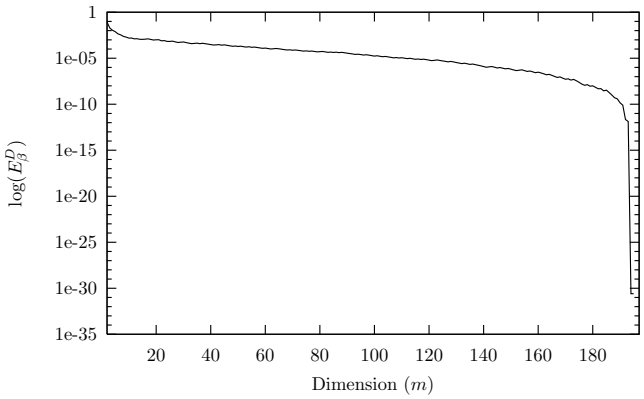
The overall best error of 0.3% was obtained by the linear support classifier in the reduced pseudo-Euclidean space of dimension 48, which was constructed using basic metric projection and the basis of the reduced space was selected using the regular technique.

3.5.2 Full Problem

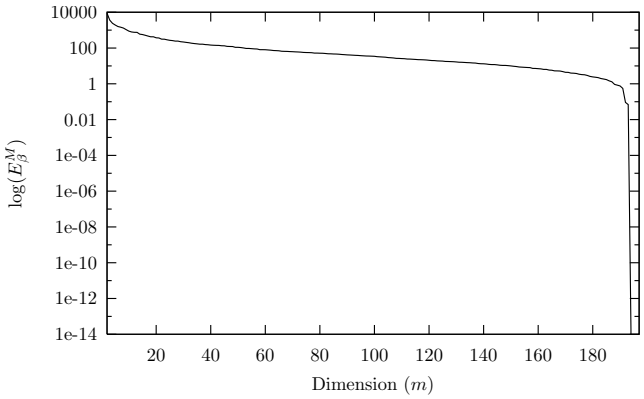
At the beginning of the experimental section it was mentioned that the symbolic dataset P we operate on corresponds to the set of phonological templates derived from the TIMIT data using a symbolic quantisation level of 10. The symbolic corpus corresponding to this quantisation level consists of 124,962 templates in the training set and 42,540 templates in the test set. The full-class task consists of evaluating the performance of vector space representation of 39 phonetic classes.

Obviously, the training set is too big to allow for the construction of a vector representation. Hence, in our experiments we make use of smaller datasets which were obtained by the symbolic clustering of the original training set. Each reduced training set has a different number of prototypes per class, from 5 up to 100 (see Section 2.6 for details). The specific symbolic algorithms we used for computations in pseudometric space (P, D_P) are the same as the ones used in a three-class problem described in Section 3.5.1. Experiments were then conducted for each of the datasets separately.

Unlike the experiments described for the three-class task, the aim of the experiments



(a) Log of the representation error E_β^D.



(b) Log of the magnitude of removed eigenvalue E_β^M.

Figure 3.11: Dimensionality reduction for the full problem: Error measures as functions of the reduced dimension m of $\mathbb{R}^{(m_+,m_-)}$. The set consists of 190 objects, 5 objects per class.

we conducted on the full class task was different. The primary goal was to test how well the pseudo-Euclidean counterpart of the symbolic k-NN AESA performs in the vector space constructed by an *isometric* (or close to isometric) embedding. This is different from the experiments in the previous section, where we tested the performance in significantly reduced vector spaces.

Clearly, an analysis similar to that performed for the small problem is still possible. For instance, Figure 3.11 (p. 115) shows the representation error measures E_β^D and E_β^M (plotted using log scale) against the dimension m of the corresponding reduced pseudo-Euclidean space. The original space corresponds to the training set of 190 objects, with 5 prototypes per class. For dimensions m higher than 10, the spaces are strictly pseudo-Euclidean.

For each of the training sets P_i with numbers of prototypes per class equal to 5, 10, 15, 20, 25, 30, 40, 45, 50, 60, 70, 80, 90 and 100, we performed the following processing steps given the corresponding pseudo-metric space (P_i, D_P):

- Determine the dimension n of the corresponding pseudo-Euclidean space corresponding to the generalised covariance matrix where all the relatively small eigenvalues (less than 10^{-4} in magnitude) are removed. For all cases, the number of removed eigenvalues corresponded to three to five eigenvalues. The resulting space $\mathbb{R}^{(n_+,n_-)}$ is nearly isometric, with the representation error being small.

- Given $\mathbb{R}^{(n_+,n_-)}$, construct two reduced vector representations in $\mathbb{R}^{(m_+,m_-)}$ (using basic δ and corrected δ_C projections of the training and test sets) without drastically reducing the space, i.e. m is close to n. In our experiments, $m = n - 1$. We employed the class-based basis selection technique for both projections.

In order to assess the performance of the vector-space version of k-NN AESA search and compare it to the symbolic counterpart, classification experiments were conducted against the full test set of 42,540 pseudo-Euclidean space vectors. In the original experiments conducted in pseudometric space (Section 2.6.5 on p. 74), the simple nearest neighbour (NN) search based on the score of the top candidate (in terms of the smallest distance to the test template) in the k-best list outperformed the majority voting schemes (i.e. optimal k was found to be 1). Therefore we decided to use the same value of the search parameter k for k-NN AESA search in pseudo-Euclidean vector spaces.

Figure 3.12 (p. 117) shows the performance of 1-NN AESA with vector representations of training and test sets constructed using basic δ and corrected δ_C metric projections. An additional curve shows the performance of the corresponding 1-NN AESA search in the symbolic pseudometric space. The accuracy curves are shown with respect to the number of prototypes per class in the respective training sets. If, for example, the number of prototypes per class is 50, then the vector space representation of the training set (obtained with either corrected or basic projection) consists of 1950 vectors.

For both basic δ and corrected δ_C constructions, the results of k-NN AESA in pseudo-Euclidean space appear to consistently outperform the pseudo-metric counterpart for all the dimensions corresponding to number of prototypes per class of up to 50. This is an indication that vector representations seem to be more robust in the sparse environment. Performance of k-NN AESA in a regular space deteriorates slightly from there on, while the k-NN AESA in a corrected vector space constructed by δ_C continues to outperform both the pseudo-metric symbolic variant and the basic vector space version constructed by δ.

The best results were obtained for 100 prototypes per class: 60.26% accuracy for 1-NN AESA on regular metric projection and 60.31% accuracy for 1-NN AESA on the corrected one. This is similar to the best result of 60.26% obtained in the pseudo-metric space.

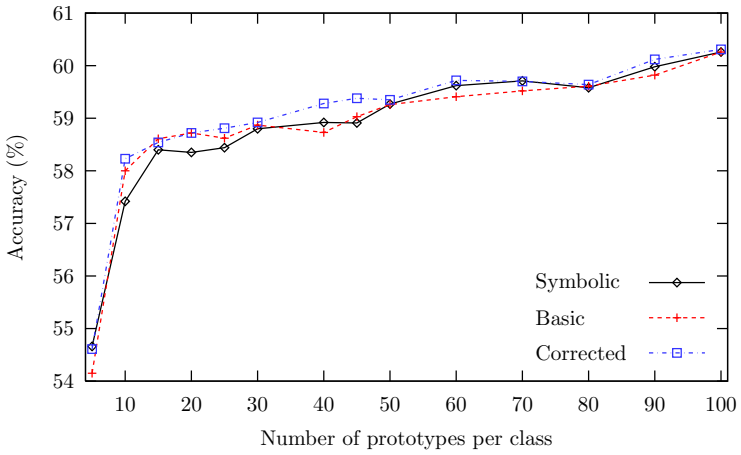


Figure 3.12: Classification accuracy (%) for the full-class problem. Performance of 1-NN AESA with vector representations constructed using basic δ and corrected δ_C metric projections is shown together with the performance of the corresponding 1-NN AESA search in the symbolic pseudometric space. The accuracy curves are shown with respect to the number of prototypes in each class.

3.6 Summary and Potential Improvements

In this chapter we examined construction of pseudo-Euclidean embeddings of finite pseudo-metric spaces representing speech. Having introduced a specific structural representation, described in Chapter 2, we performed a dissimilarity-based transition to a more analytically developed vector space representation, which for our particular task is pseudo-Euclidean. This pseudo-Euclidean vector space preserves the isometric properties of the original symbolic phonological metric space. We used the resulting space to experiment with the more efficient vector-space counterparts of the symbolic classifiers, perform data analysis and construct linear decision surfaces (which are otherwise unavailable to us in symbolic spaces) for multi-class problems. On the theoretical side, we explored several metric projection and basis selection algorithms. In particular, in Section 3.4.2 we focused on the corrected metric projection technique proposed by Goldfarb (1986), which otherwise is not treated in his earlier monograph (Goldfarb, 1985). We also suggest a new algorithm for optimal basis selection of the reduced pseudo-Euclidean spaces which, in some cases, outperforms the classical approach.

When the original (pseudo) metric separates the classes reasonably well (as is the case with a three-class TIMIT problem, which was examined first), a wide array of linear (and otherwise) hyperspace classifiers consistently outperform both the symbolic algorithms and the numeric counterparts of the symbolic algorithms based on the Nearest Neighbour rule. These findings agree with the results of the other dissimilarity-based vector space representation experiments previously reported in the theoretical pattern

recognition literature (Duin *et al.*, 2004; Graepel *et al.*, 1999; Pękalska, 2005; Pękalska and Duin, 2002; Pękalska *et al.*, 2002).

We also conducted full classification experiments on the 39-class TIMIT task in order to verify the hypothesis that better results can be obtained with the generalised version of the k Nearest Neighbour AESA search in the constructed isometric pseudo-Euclidean spaces. We confirmed this hypothesis, obtaining a small improvement in 1-NN AESA search (in a space constructed using the corrected metric projection) over its symbolic counterpart. Overall, the results of the experiments on a full task are very close (and in most cases superior, especially when the corrected metric projection technique was used) to the results in the original pseudo-metric space.

Potential Improvements

Since the future directions of our research will include more structurally complex and hence less computationally efficient representations of spoken language, such as graphs (Bird and Liberman, 2001), reliable construction of robust pseudo-Euclidean space representations of the pseudo-metric spaces in question seems to us to be of paramount importance. The theory of dissimilarity representations, pioneered by Goldfarb (1979; 1984; 1985), is becoming more popular (Bicego *et al.*, 2004; Duin and Pękalska, 2005; Duin *et al.*, 2004; Pękalska, 2005). Several important open problems, however, remain unresolved. Here, we briefly enumerate some of the more important ones, which are of direct relevance to us:

Clustering in pseudo-Euclidean vector space:

In our experiments, we used the phonological template datasets which were pre-clustered in the original pseudo-metric space (P, D_P) using symbolic techniques. We believe that this is definitely suboptimal since better clustering algorithms are available in numeric vector spaces. For instance, the concept of a mean of a set is naturally supported by the underlying vector space and can be computed in a negligible amount of time. The same applies to other analytical machinery, like multivariate analysis which is simply missing in the symbolic spaces.

The reason why no clustering was attempted directly in pseudo-Euclidean space is connected with the complexity of the overall task. Recall that the structural training set of phonological templates corresponding to TIMIT has 124,962 objects. In order to perform reasonable analysis of the covariance of this training data, the naive approach is to embed the entire training dataset into the pseudo-Euclidean space, which of course is not computationally tractable. Note that most of the datasets reported in the literature are numeric and therefore can be efficiently pre-clustered. In addition, these datasets

have a significantly lower complexity than TIMIT (in terms of the number classes under investigation and their separability), hence the issue of handling large datasets may have not been crucial.

Briefly, in order to better tackle the issue of more efficient reduction of the training data, we will need to devise hybrid techniques which somehow operate in both pseudo-metric space (defined by dissimilarities only) and pseudo-Euclidean (numeric) space by assembling the candidate clusters iteratively and assessing the measure of their goodness in pseudo-Euclidean spaces. One of the concepts which may potentially help us to gain more insights into this process is the concept of *pseudo-metric decision trees* suggested by Goldfarb (1986). Although this concept might not be directly useful, the idea that the dissimilarity-space can be hierarchically split into several subspaces, each of which is optimised separately, can guide the development of new clustering algorithms for large dissimilarity-based datasets.

Mathematically justified classifiers:

Despite the fact that linear support vector classifiers performed well on a reasonably separable three-class task, there seems to be a problem with the approach we followed. The experiments on the three-class task were based on discarding (or distorting) the geometry of the pseudo-Euclidean space, essentially turning it into Euclidean, similar to the assumptions described by Graepel *et al.* (1999) and Pękalska *et al.* (2002). In other words, in latter works the authors constructed the classifiers based on the Euclidean assumptions (the positive definite form of the space's matrix of symmetric bilinear form), treating the original pseudo-Euclidean space $\mathbb{R}^{(n_+,n_-)}$ as a regular euclidean space $\mathbb{R}^n$, where where $n = n_+ + n_-$. Recently Haasdonk (2003) and Haasdonk and Bahlmann (2004) attempted to arrive at a different formulation of the learning problem where the classifier is generalised to operate in a pseudo-Euclidean setting. In particular, Haasdonk (2003) offers a theoretical analysis of the performance of support vector machines in terms of convex hulls in pseudo-Euclidean space and proposes a new algorithm for discovering the optimal hyperplane with support vector classifiers in that space. Unfortunately, at present we are not aware of any reports of experimentation with this variety of support vector classifiers.

Chapter 4

Inductive Learning with ETS_0

4.1 Introduction

So far we have discussed two techniques for modelling speech. In Chapter 2 and in (Gutkin and King, 2004b) we presented a symbolic template-based representation of speech based on phonological distinctive features. As we saw, the chief benefit of that representation is that it is amenable to linguistic analysis. Furthermore, we showed how to derive it from real speech data and conduct classification. One of the main drawbacks of the approach presented is that the set of analytical tools available for modelling in symbolic spaces (in our case corresponding to the phonological pseudo-metric space) is quite restricted when compared to the wide array of techniques that conventional vector spaces offer. In addition, we did not discuss the possible ways of learning in the phonological pseudo-metric spaces, apart from mentioning (very briefly) in Section 2.7 the possibility of optimising the numeric weights on the similarity measures defined in those spaces.

In Chapter 3, which followed, and in (Gutkin and King, 2004a), we discussed the numeric framework based on dissimilarities, which allows for flexible integration of the above symbolic representation with vector spaces. We also mentioned that the main motivation for this integration was a need to find a class of vector spaces (which turned out to be pseudo-Euclidean) which optimally represented the phonological pseudo-metric space at hand. The optimality criterion for the transition from the symbolic to vector space was naturally defined to preserve the dissimilarities between the objects in the original symbolic and target vector spaces. Once this transition was accomplished, the chief benefit of the vector spaces — availability in these spaces of several efficient learning and classification techniques (like neural networks and support vector machines) — could be utilised. Two problems can easily be identified with this approach, however. First of all, once the representation is transferred to (any) vector space, the linguistically

expressive power of the original symbolic representation is lost, since the new representation is numeric. In addition, since the transition is purely dissimilarity-based, there is not enough information to perform an inverse mapping back to the symbolic space[1]. Unfortunately, this one-way transition is a limitation of all multi-dimensional scaling and embedding approaches. Thus, it appears that the original hopes for impending unification of the symbolic and numeric approaches to pattern recognition (Goldfarb, 1984), following the introduction of the theory of the isometric pseudo-Euclidean space embeddings, were rather premature.

Coming back to the discussion of symbolic spaces and phonological pseudo-metric space in particular, we are now in a position to consider the issue of learning in the symbolic environment which has been tacitly omitted from the exposition up until now. Earlier in this thesis we mentioned that the main analytical tool of the symbolic space, namely the dissimilarity measure, possesses a numeric component which is expressed in terms of weights defined on the associated edit operations (see Section 2.4). We also mentioned that one could attempt to define learning in the phonological pseudo-metric space in terms of the optimisation of the numeric weights of the corresponding dissimilarity measure. Such an optimisation, however, is not attractive. On the one hand it does not provide us with any means of discovering interesting *symbolic* information. On the other, it forces one to remain in the symbolic modelling space, preventing access to the sophisticated vector space machinery, which otherwise is available if one decides to do all the optimisation in the corresponding vector space. Hence, we believe that if the aim of the learning is pure weight optimisation, it is more advantageous to transfer the problem to a pseudo-Euclidean domain using the techniques covered in Chapter 3 (or employ any other alternative dissimilarity-based transition, along the lines suggested by Pękalska, 2005) and use more powerful tools for learning and classification in these spaces. This was our main consideration for omission of the learning process in the symbolic spaces from the discussion.

In this chapter, we address the issue of learning in the phonological pseudo-metric spaces. The particular approach we pursue and some of the experimental results were previously reported by us in (Gutkin and King, 2005b). Informally, we can approach the learning problem by spelling out the following requirements, which are the main objectives of the work we describe in this chapter:

- The phonological representation defined in Chapter 2 does not provide us with any means of *class description*. In the rest of this chapter, we will refer to such representations as "rigid". Normalised edit distance between the phonological tem-

[1] Having learnt a class representation in a dissimilarity space, one may, perhaps naïvely, hope to somehow perform an inverse mapping back to the symbolic space, where this class information can be interpreted better.

plates, for instance, does not furnish us with any understanding of the *structural* makeup of the particular class of phones in question. Therefore, we would like the phonological representation to provide us with the means of encoding the compositional makeup of the phonological classes being modelled.

- Moreover, even if we learn the optimal weights for the dissimilarity measures from the training data (thus achieving better separation between the classes), we would still not be able to learn anything about *what* makes the phones *structurally* different. Hence, we would also like to have the means of *learning* the structural class descriptions from the data.

Since we are dealing with real speech classes, which do not have any closed form description (expressed in terms of some syntactic grammar, for instance), out of necessity learning model has to combine in itself both structural and numeric components. The numeric component is needed to ensure that some optimisation objectives are met. In fact, syntactic (i.e. grammar-based) approaches to pattern recognition have been repeatedly criticised precisely because most real world learning problems cannot be described by a purely syntactic grammatical approach (see, for example, remarks by Watanabe (1985), Tanaka (1995) and Pavlidis (2003)). Although the model we consider in this chapter is different from the syntactic grammar-based approaches (as will be shown below), we nevertheless take these remarks into account.

The above requirements can be more formally cast as the following *Inductive Learning* problem (Abela, 2001; Goldfarb and Nigam, 1994):

Definition 4.1 (Inductive Learning Problem)**.** Given a finite set C^+ of positive training objects that belong to a (possibly infinite) set C (concept) to be learnt and a finite set C^- of negative training objects that do not belong to the concept C, automatically construct a class representation for C based on negative and positive training objects and, *as a consequence*, to recognise if a new element belongs to C.

The *structure of a class* is taken to be:

1. The *symbolic* features[2] that make the objects of the same class similar to each other and/or different from other objects outside the class.

2. The *emergent* combinatorial interrelationships among these features.

The inductive learning process would then involve the discovery and encoding of the structure of the class allowing one to abstract (generalise) and associate meaning with the set of objects. In the subsequent recognition stage, the *induced* dissimilarity measure is used to compare a new object to some fixed and reduced set of objects from C^+. □

[2]We refer to this notion in pattern recognition terms. It is not to be confused with distinctive phonological features from Chapter 2.

The Evolving Transformation System (ETS$_0$) formalism has been initially developed by Goldfarb (1990; 1992) to address the above requirements[3]. In particular, ETS$_0$ has been successfully used by Abela (2001) in a grammatical inference setting and was the basis for the inductive mathematical model of human vision proposed by Goldfarb *et al.* (1996). It should be noted that ETS$_0$ is not an algorithm, but rather a model, providing formal guidelines which specify how to address the inductive learning problem posed above given any symbolic pseudo-metric domain. For our task, where the objects we model are the phonological feature templates, we augment (using the ETS$_0$ model) the existing phonological pseudo-metric space representation (specified in Chapter 2) in such a way as to meet the objectives of the inductive learning problem.

One of the central ideas of the ETS$_0$ formalism is that the similarity measure plays the critical role in the definition of a class (Goldfarb, 1992) via capturing the compositional makeup of objects. Learning in ETS$_0$ essentially reduces to finding a distance function (defined in terms of a set of weighted transformations) that achieves some degree of class separation by minimising the distance between the objects in the positive training set C^+ while at the same time making sure that the distance between the objects in C^+ and the objects in the negative training set C^- is always greater than some non-zero positive threshold. An ETS$_0$ learning algorithm achieves this by iteratively modifying the distance function in order to achieve the above objectives, hence we can call this distance function an *evolving dissimilarity measure*. Representation of a class C in ETS$_0$ is thus defined in terms of a set of some prototype objects belonging to C^+ and the non-empty set of non-trivial structural transformations. These transformations, discovered during learning, play the role of the structural features providing the makeup of the class C. The classification procedures, described in Chapter 2, that employ the evolving dissimilarity measures can be used as usual. The only difference is that the dissimilarity measure is now conceptually more interesting.

In this chapter, we describe the learning algorithms defined in the phonological pseudo-metric space we use, along with several optimisation criteria. The learning procedure allows one to discover linguistically meaningful compositional makeup of various classes of the phonemes under investigation. In addition, we describe experiments which were conducted to verify the hypothesis that the phonological representation which is designed according to the principles outlined in Definition 4.1 results in the improved phoneme classification performance.

[3]The subscript 0 refers to the initial version of the formalism, which has undergone significant development since its inception. The newer versions of the formalism, however, are not suited to the phonological pseudo-metric space modelling we consider here. They are described in the following chapters.

Overview of the chapter

The necessary background material which can be helpful to better understand the basic motivation behind the development of the ETS_0 model (Goldfarb, 1990), as well as generalisations of some important notions from Chapter 2, is presented in Section 4.2. Next, we introduce the ETS_0 model in a general setting in Section 4.3. This section continues to provide a running example of a toy phonological representation that was first introduced in Example 2.3. Section 4.4 describes the experimental setup (compatible with the setup of the experiments reported in Chapters 2 and 3) along with the discussion of the results, which are compared against the results reported in the previous chapters. We summarise the chapter in Section 4.5 and present some future directions of research which will potentially improve the learning algorithms presented in this chapter.

4.2 Preliminaries: Objects, Transformations and Metrics

This section is setting the scene (rather informally) for subsequent developments. Let S represent the set of *homogeneously* structured objects. For example, S may be a collection of rooted binary trees or strings. In his paper, Goldfarb (1990) suggested to represent the objects in S by graphs. He showed how using graphs allows one to treat various types of objects using a single underlying formal structure without loss of generality. Given any object s in S, its structural representation as a graph can be considered as *whole*, while any fragment (sub-graph) of this representation can be seen as *part*. Next, Goldfarb (1990) showed how the concept of structural dependencies between the objects in S can be elaborated by introducing *transformations* on the objects. Informally, the transformation operation allows to modify the structure of an object s in S by replacing, deleting or substituting any of its parts, obtaining a new object s'.

The concept of transformation allows to introduce the notion of similarity on homogeneously structured objects. The notion of similarity, in turn, allows the definition of non-parametric and parametric distances between the structural objects. These issues are treated in Section 4.2.1, where the exposition is based on (Goldfarb, 1990, 1992).

In Section 4.2.2 we describe the weighted string edit distance algorithm which employs arbitrary string transformations (not confined to trivial single-character operations). This algorithm has been developed by the author and is extensively used in our representation, which, being a generalisation of the framework described in Chapter 2, is based on string templates.

4.2.1 Structural Dissimilarity Measures

Given the two objects s_1 and s_2 belonging to some set S, it is reasonable to assume that either s_1 can be transformed into s_2 or vice versa. In particular, given the above notion of the structural transformation, it is possible to introduce the notion of *similarity* between the two structural objects via the transformation concept (Goldfarb, 1990):

Definition 4.2 (Similarity)**.** Let O denote the universe of possible transformation operations defined on structural objects in S. Object $s_1 \in S$ is *similar to* object $s_2 \in S$ if there exists a finite sequence (chain) of m transformation operations $O' = \{o_1, o_2, \ldots, o_m\} \subseteq O$ such that when applied to s_1 it results in s_2. □

Goldfarb (1990) showed that any sequence of operations that transforms one object into another is *reversible*, which means that this sequence can be reversed by reversing each of the constituent transformation operations. In addition, he introduced the notion of *completeness* for a set of transformations O defined on a set of objects S. A set O of transformations is *complete* for the set S, if any pair of objects $s_1, s_2 \in S$ are similar.

For most of the classes of structures (vectors, strings, trees, etc.) one can always choose the set of appropriate "insertion-deletion" operations which make the set of operations complete. Perhaps the most obvious example are the strings over some finite alphabet. Any string can be obtained from another by applying a sequence of insertion and deletion single-character transformations. If, on the other hand, one restricts the set of transformations to only include the insertion operations, then the completeness criterion is violated.

Given the notion of similarity between the two objects, there is a natural way of introducing a metric on the set of objects S. The metric measure is defined via the set of the transformations O, turning the pair (S, O) into a metric space (Goldfarb, 1990):

Definition 4.3 (Intrinsic Distance)**.** Let O denote the set of transformations which are complete over the set of objects S. The (structural) *intrinsic distance* between any two similar objects $s_1, s_2 \in S$ in (S, O) is the function Δ_O defined on S as

$$\Delta_O \colon S \times S \to \mathbb{N}, \;\; \mathbb{N} = \{0, 1, 2, \ldots\},$$

where Δ_O is the minimum number of transformation operations necessary to transform s_1 into s_2. □

The adjective "intrinsic" in the above definition, coined by Goldfarb (1990), refers to the fact that the distance function Δ does not reflect any *empirical (statistical)* knowledge about the transformation operations. Such knowledge can only be acquired within the learning framework, which is introduced later on in this chapter. A possible

intrinsic distance function for the class of strings over a finite alphabet (which, as we have seen above, is complete under insertion and deletion single-character transformations) is a Levenshtein edit distance (Levenshtein, 1966; Sankoff and Kruskal, 1983).

The flexibility of the intrinsic distance function can be improved by allowing different transformation operations to have different weights associated with them. These weights, for instance, may express some *a priori* domain-specific knowledge about the transformations involved. For example, we can consider some of the transformations to be more important than others by assigning to them larger weights. This particular distance function is called the *parametric distance*, defined below (Goldfarb, 1990, 1992):

Definition 4.4 (Parametric Distance)**.** Let S be the set of structural objects and O the set of m transformation operations defined on the objects in S. *Parametric distance function*

$$\Delta_\omega \colon S \times S \to \mathbb{R}^+$$

defined for the pair (S, O) is obtained from the intrinsic distance function Δ as follows:

- Each of the m transformation operations o_i in O is assigned a corresponding weight ω_i, which is a non-negative real number.

- For any pair of objects s_1 and s_2 in S let

  $$g = \left\{(o_1^g, \omega_1^g), (o_2^g, \omega_2^g), \ldots, (o_k^g, \omega_k^g)\right\}$$

  denote a sequence of k operations together with their corresponding weights required to transform s_1 into s_2. Also, let G denote the set of all such transformation sequences between s_1 and s_2.

- The distance between any pair of objects s_1 and s_2 in S is defined as

  $$\Delta_\omega(s_1, s_2) = \min_{g \in G} \sum_{i=1}^{|g|} \omega_i^g \,, \tag{4.1}$$

  where the minimum is taken over the set of all possible sequences of operations G transforming s_1 into s_2. □

The above definition can be seen as a generalisation of the weighted edit distance for strings, which we discussed in Section 2.4.2 of Chapter 2. As noted by Abela (2001), the above definition of the parametric distance can be made even more general by allowing the individual structural transformations to be assigned real multi-dimensional vectors of weights, rather than single weights. This is potentially useful if one wants to use semantically different numeric parameters (such as similarity cost, penalty and

deletion cost) for each individual transformation. In addition, instead of computing the parametric distance as an optimal cost sequence using the sum in equation (4.1), some other criterion ϕ can be potentially chosen. In other words, one can perform a different optimisation

$$\Delta_\omega(s_1, s_2) = \min_{g \in G} \phi(\omega^g)\,,$$

where ω^g is the weight vector corresponding to the transformation sequence g. The two modifications discussed above are outside the scope of this chapter, however.

Given the above notions of a structural object representation, transformations and the similarity measures, the Evolving Transformation System model is finally introduced in Section 4.3. As we shall see, the ETS$_0$ representation we consider is a generalisation of the phonological metric spaces which were discussed in Chapter 2. In the discussion above we mentioned that the set of operations needed for transforming the structural objects into each other is not confined to trivial operations only. Thus the notion of intrinsic distance from Definition 4.3 is not as simple as it may seem.

Before proceeding with the exposition of ETS$_0$, in the next section we introduce the intrinsic distance function which has been developed by us for distinctive feature streams (modelled as strings) from the phonological representation in Chapter 2.

4.2.2 Block Edit Distances on Strings

Block edit distances are recent alternatives to character edit distances. Such distances introduce block-based edit operations in addition to the character-based ones and can be described in terms of the minimum number of single character and block edit operations required to transform one string into another. In this work, we developed a block edit distance algorithm which can be seen as a generalisation of a classical (weighted) Levenshtein edit distance. This distance function is useful to us because it allows computation of dissimilarities on strings that use an arbitrary set of string transformations. As we shall see in Section 4.3, the non-trivial blocks play the role of the *features* (in a sense of Definition 4.1) that are discovered during the learning in phonological ETS$_0$ representation.

The conventional counterparts of the block distances were discussed in Section 2.4.2, which also introduced the necessary notation which is followed in the exposition below.

4.2.2.1 Generalised String Edit Problem

Similar to the single-character based metrics of Section 2.4.2.1, given the two strings

$$A = a_1, a_2, \ldots, a_n \quad \text{and} \quad B = b_1, b_2, \ldots, b_m$$

over some finite alphabet Σ, the aim is to compute the edit distance between A and B. An additional symbol, not belonging to an alphabet Σ, is an empty string denoted by ϵ. By a *block* we mean a (possibly empty) string over Σ. We next generalise the concept of an edit operation, originally given in Definition 2.5.

Definition 4.5 (Block Edit Operation)**.** An *edit operation* e is an ordered non-empty pair (F_i, F_j), where F_i and F_j are the blocks over Σ. Note that the pair is non-empty, hence at least one block in the pair has to differ from ϵ.

String B results from string A via (F_i, F_j) if

$$A = S_1 F_i S_2 \quad \text{and} \quad B = S_1 F_j S_2$$

for some strings S_1 and S_2 over Σ. The pair (F_i, F_j) is called a *replacement* if $F_i \neq \epsilon$ and $F_j \neq \epsilon$, a *deletion* if $F_j = \epsilon$ and an *insertion* if $F_i = \epsilon$. □

Let

$$I = \left\{(F_i, F_j)\right\}, \quad |F_i| > 1 \quad \text{and/or} \quad |F_j| > 1$$

denote the set of non-trivial blocks, which in practice is supplied as an input to the algorithm. Let F denote the set of block operations. Hence, the set of all trivial operations is given by $F \setminus I$. In the case when the set I is empty, the problem reduces to the conventional string edit distance considered in Section 2.4.2. Next, the notions of an edit sequence (Definition 2.6) and a cost function (Definition 2.7) can be generalised to the block edit problem:

Definition 4.6 (Block Edit Sequence)**.** A sequence E of block edit operations is called a *block edit sequence*. Let

$$E = e_1, e_2, \ldots, e_k$$

be an edit sequence. B is said to be *derivable* from A if there exists a sequence of strings $S_0, S_1, \ldots, S_k$ such that $A = S_0, B = S_k$ and for $1 \leq i \leq k$, S_i results from S_{i-1} via e_i. B is always derivable from A via a sequence consisting of n trivial deletion and m trivial insertion operations. □

Definition 4.7 (Block Edit Cost Function)**.** A *cost function* δ is a binary mapping assigning a non-negative real number to each block edit operation (F_i, F_j). Thus, the cost of a sequence E of length k is given by

$$\delta(E) = \sum_{i=1}^{k} \delta(e_i)\,. \qquad \square$$

4.2.2.2 Generalised Wagner-Fisher Algorithm

In order to compute a classical weighted Levenshtein distance between the two strings one can use an efficient Wagner-Fisher algorithm (Section 2.4.2.2). We modified this algorithm in a reasonably straightforward way to address the *generalised* weighted Levenshtein case. This novel algorithm, which we call the *Generalised Wagner-Fisher Algorithm*, employs a simple dynamic programming approach based on its classical version and is described below.

Let

$$A_{i,j} = a_i, a_{i+1}, \ldots, a_j \quad \text{and} \quad B_{i,j} = b_i, b_{i+1}, \ldots, b_j \,.$$

denote two substrings of A and B, respectively. Also let

$$A_i = a_1, a_2, \ldots, a_i \,, \quad B_j = b_1, b_2, \ldots, b_j \,, \quad \delta_{i,j} = \delta(A_i, B_j) \,.$$

Construct a $(n+1) \times (m+1)$ matrix

$$D = (d_{i,j}) \quad i \in [0, n],\ j \in [0, m] \,.$$

Initially $d_{0,0} = 0$, the first column is given by

$$d_{i,0} = \delta(A_i, \epsilon) = \min_{e \in F} \{\, d_{i-k,0} + \delta(e) \,|\, e = (A_{i-k,i}, \epsilon),\ 1 \leq k \leq i \,\}$$

and the first row by

$$d_{0,j} = \delta(\epsilon, B_j) = \min_{e \in F} \{\, d_{0,j-k} + \delta(e) \,|\, e = (\epsilon, B_{j-k,j}),\ 1 \leq k \leq j \,\} \,.$$

For the rest of the elements,

$$d_{i,j} = \delta_{i,j} \,,$$

where $\delta_{i,j}$ is calculated using the following recursive relation that can be seen as the generalisation of Wagner and Fisher's result (Wagner and Fisher, 1974):

$$d_{i,j} = \min_{e \in F} \begin{cases} \{\, d_{i-k,j} + \delta(e) & | \quad e = (A_{i-k,i}, \epsilon)\,, 1 \leq k \leq i & \}\,, \\ \{\, d_{i,j-k} + \delta(e) & | \quad e = (\epsilon, B_{j-k,j})\,, 1 \leq k \leq j & \}\,, \\ \{\, d_{i-p,j-q} + \delta(e) & | \quad e = (A_{i-p,i}, B_{j-q,j})\,, 1 \leq p \leq i\,,\ 1 \leq q \leq j & \} \end{cases}$$

for $i \in [1, n]$ and $j \in [1, m]$. The resulting block edit distance $\delta(A, B)$ is stored in $d_{n,m}$.

The algorithm is slower than its basic single-character based version because of the additional iteration over the set of operations and extensive string matching. It uses $O(n \cdot m \cdot l)$ elementary steps, where $l = |F|$ is the cardinality of the set of block and character operations.

One can improve the string matching performance by the use of suffix trees (Gusfield, 1997). The overall time complexity of the algorithm could be further reduced

by applying the *t-blocks* method as used by Masek and Peterson (1980; 1983) in their research aimed at speeding up the classical weighted Levenshtein distance. The t-blocks is a speedup technique for dynamic programming (known as the Four-Russians Method) first suggested by Arlazarov *et al.* (1970). It uses *t-blocks* (t by t squares in the dynamic programming table) rather than single cells at each step of the Wagner-Fisher algorithm.

4.3 Evolving Transformation Systems (ETS$_0$) Model

In this section we introduce the core components of the Evolving Transformation System (ETS$_0$) model. The basic structure of the model is provided by the *transformation system*, described in Section 4.3.1, which can be seen as a certain generalisation of the notion of the pseudo-metric space in which there is more emphasis on the *structure* of the objects in the domain. Next, we outline (rather informally) the learning process which in ETS$_0$ is seen as sequential optimisation of the transformation system structures. This is covered in Sections 4.3.2–4.3.3. Finally, the goal of the learning process is the discovery of *inductive class structure*, described in Section 4.3.4.

It is important to note that the algorithmic parts of this section are primarily based on previously reported ETS$_0$ findings, which have been described with varying degree of detail by Goldfarb (1990, 1992); Goldfarb *et al.* (1995); Goldfarb and Deshpande (1997); Goldfarb *et al.* (1996); Goldfarb and Nigam (1994), Kamat (1995) and Abela (2001).

4.3.1 Transformation System

Given the notion from the previous section, the concept of *transformation system* can be introduced in a straightforward manner (Abela, 2001; Goldfarb, 1990, 1992; Goldfarb *et al.*, 1996):

Definition 4.8 (Transformation System)**.** A *transformation system* (TS) is a 3-tuple $T = (S, O, D)$, where

- S is a set of structural objects of the representation;
- O is a finite set of m substitution operations for transforming the objects in S satisfying the following two trivial properties: all the operations in O are reversible and O is complete (Section 4.2.1);
- The set $D = \{\Delta_\omega\}$ is a *family of parametric distance functions* defined on (S, O) (parametric distance functions are given in Definition 4.4).

The number of possible parametric distance functions in D is essentially dictated by the weighting scheme Ω, which is a set of all the possible weight vectors ω of dimension m (corresponding to m transformations in O) which possess the following property:

$$\forall \omega \in \Omega \colon \sum_{i=1}^{m} \omega_i = 1 \,. \qquad \square$$

We can refer to D as a family of *competing* distance functions. The set of transformations O is complete, therefore for any pair of objects s_1 and s_2 there exists *at least* one sequence of transformations which will transform s_1 into s_2. In practice, one usually finds such a sequence together with the appropriate weight vector ω in Ω which minimises $\Delta_\omega(s_1, s_2)$. Hence the use of the adjective "competing" above.

In Section 2.4 we defined the phonological metric space — the distinctive feature-based speech representation. The phonological metric space is defined as the set of all phonological templates in the domain, denoted $\mathbb{P}$, together with a real-valued mapping $d_{\mathbb{P}}$ defined on this set $\mathbb{P}$ (Definition 2.4). As we saw in Chapters 2 and 3, this real-valued mapping can be metric, semimetric or pseudo-metric. Below, we propose to interpret this phonological representation as a transformation system:

Definition 4.9 (Phonological Transformation System)**.** A *phonological transformation system* is a 3-tuple $T_{\mathbb{P}} = (\mathbb{P}, O_{\mathbb{P}}, D_{\mathbb{P}})$, where $\mathbb{P}$ is the set of all phonological templates from Definition 2.4, $O_{\mathbb{P}}$ is the set of all possible distinct transformations which transform any pair of templates from $\mathbb{P}$ into each other and $D_{\mathbb{P}}$ is a family of distance functions on the set of phonological templates from Definition 4.8, which is defined via $O_{\mathbb{P}}$. $\square$

Note that this extension is straightforward, since all dissimilarity measures on phonological templates we considered in the phonological metric space (see Section 2.4) possess one important property which eases the transition: all the metrics we considered are defined in terms of transformations. The only conceptual differences are the following:

- The set of transformations $O_{\mathbb{P}}$ is not restricted to the single-character operations. Hence, the phonological metrics $d_{\mathbb{P}}$ can be defined in structurally more interesting ways.
- The introduction of a general requirement on the weighting scheme of the distance functions $d_{\mathbb{P}}$ which belong to the parametric family $D_{\mathbb{P}}$.

More importantly, instead of one phonological metric space, we now have a *collection* of phonological metric spaces. In other words, phonological transformation system $T_{\mathbb{P}}$ can be represented as a following set

$$T_{\mathbb{P}} = \left\{ \mathbb{P}, O_{\mathbb{P}}, \Delta_{\mathbb{P}}^{\omega} \right\}, \quad \omega \in \Omega$$

where different metrics $\Delta^{\omega}_{\mathbb{P}}$ correspond to different weighting schemes in Ω.

The similarity measure $d_{\mathbb{P}}$ between any two phonological templates p_1 and p_2 (each consisting of N distinctive feature streams) is expressed in terms of a linear combination of N per-stream similarity measures d_i, $1 \leq i \leq N$. Since the streams (modelled as strings) are independent of each other, so are the sequences of transformations which transform them. Hence, one can define the weighting scheme on these transformations to be stream-specific. The phonological transformation system $T_{\mathbb{P}}$ can therefore be seen as a set of N independent *distinctive feature transformation systems* T^i, $1 \leq i \leq N$, one for each stream.

Example 4.1. Figure 4.1 (p. 133) shows the three stream-specific phonological transformation systems corresponding to a toy phonological template representation of the two phonemes [p] and [b] from Example 2.3 (Figure 2.5). Two instances of each phoneme are shown in the figure. The first transformation system (T^1) corresponds to the [consonantal] stream, the second (T^2) to the [sonorant] and the third (T^3) to the [tense] stream. The transformation system-specific weights are not shown. Each transformation system is a string transformation system employing its own objects, transformations and parametric distance functions.

The set of objects S^1 in T^1 transformation system, for example, consists of four instances of [consonantal] stream encountered in the four templates shown on top of the figure. The set of transformations O^1 in T^1 consists of three single character operations. The weighting scheme Ω^1 corresponding to O^1 may assign an equal normalised weight $\frac{1}{3}$ to each of the transformations. ▷

4.3.1.1 Learning in the Transformation System

The inductive learning problem was postulated in Definition 4.1. Suppose that during the learning stage the amount of information available to the system is restricted to a finite set of labelled phonological templates from each of the M disjoint classes given by

$$C = \{C_1, C_2, \ldots, C_M\} = C^+ \cup C^- ,$$

where

$$C_i = \{p^i_1, p^i_2, \ldots, p^i_{k_i}\}$$

is a set of k_i training templates representing some class. We refer to the set C as the *global training set*. Suppose that a class to be learnt is C_i, i.e. it is a class of positive objects C^+. Then the set of negative training objects C^- is given by $C \setminus C^+$.

Let C^+ and C^- be the sets of positive and negative training templates from some finite labelled set. In Definition 4.8 it was mentioned that the set of transformations O

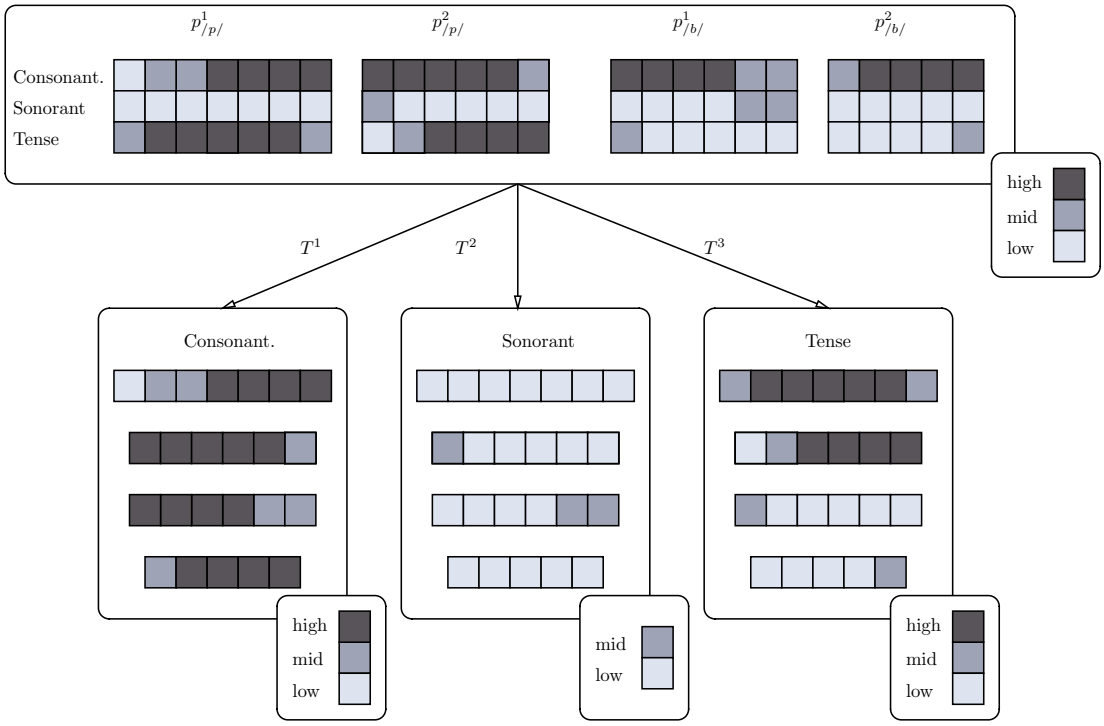


Figure 4.1: Stream-specific phonological transformation systems T^1, T^2 and T^3 corresponding to the phonological SPE template representation from Example 2.3 shown in Figure 2.5. For each transformation system, the corresponding sets of objects and single-character transformations are shown. The weighting scheme is not shown.

corresponding to T is fixed. Therefore the only possible way of learning in transformation systems is numeric. The learning is performed by finding some optimal weighting scheme $\hat{\omega}$ corresponding to the transformations O. This is achieved by optimising the weight function $f(\omega)$ of the following form:

$$f\colon \mathbb{R}^m \to \mathbb{R}\,,$$

where m is the number of operations in O. In all the studies which address the issue of learning in transformation systems (Abela, 2001; Goldfarb, 1990; Goldfarb *et al.*, 1995; Goldfarb and Nigam, 1994), the following generic (numeric) optimisation criterion is used:

$$\max_{\omega\in\Omega} f(\omega) \;= \max_{\omega\in\Omega} \frac{\beta(\omega)}{\epsilon + \alpha(\omega)}\,. \tag{4.2}$$

In the equation above, $\beta(\omega)$ is the *degree of separation* between the positive and negative training sets C^+ and C^-, $\alpha(\omega)$ is the degree of separation between the positive training objects in C^+ and ϵ is a small positive constant to prevent the overflow condition when the values of $\alpha(\omega)$ approach zero. Both $\alpha(\omega)$ and $\beta(\omega)$ are defined in terms of parametric distance function Δ_ω. The optimisation function $f(\omega)$ combines in itself both the measure of compactness of C^+, as well as the measure of separation of C^+ from C^-, following from the simultaneous minimisation of function α and maximisation of function β.

Hence, the learning in a transformation system reduces to finding a distance function $\Delta_{\hat{\omega}}$ (parametrised by the weight scheme $\hat{\omega}$) that achieves some satisfactory class separation. In other words, a measure such that at the end of the learning process there is a high degree of proximity between the objects in C^+ (the distance between the objects is close to zero), while the distance between the objects in C^+ and C^- is non-zero. Thus, at each iteration of the learning algorithm, the candidate weight scheme brings the objects in C^+ closer to each other, while maintaining some reasonable distance between them and the objects in C^-. The learning stops when all the members of C^+ are in some small neighbourhood δ. This process in depicted in Figure 4.2. Once the learning is complete, the set of all templates P that belong to a concept C represented by C^+ can be defined to be

$$\left\{p \in P \,|\, \Delta_{\hat{\omega}}(p, \hat{p}) < \delta\right\},$$

where $\hat{p} \in P$ is an object in the δ-neighbourhood chosen as a centroid (or *attractor*, according to Abela, 2001).

The result of the optimisation process is the set of optimal vector weights

$$\hat{\omega} = \arg\max_{\omega\in\Omega} f(\omega)$$

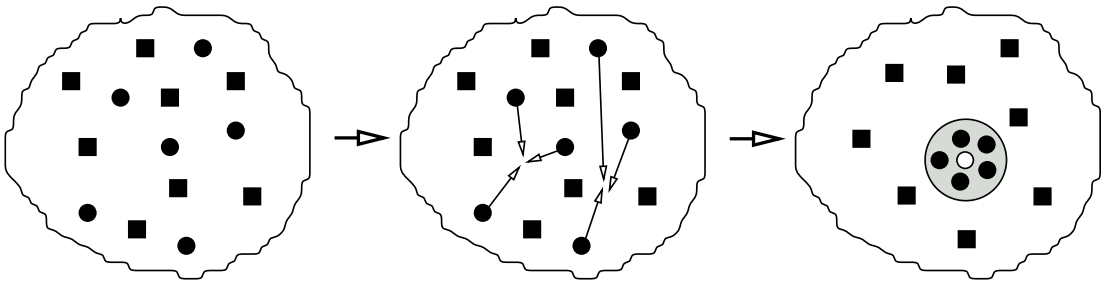


Figure 4.2: Depiction of the learning process within ETS$_0$. The domain is represented by the positive C^+ (black circles) and negative C^- (black squares) training sets, shown on the left-hand side of the figure. During learning, the members of C^+ move closer to each other (shown in the middle of the figure), until they all end up in a small neighbourhood (shown shaded on the right-hand side of the figure). In order to define this neighbourhood, one selects any member of a positive training set as a *centroid* (shown as a white circle). For a transformation system the learning is numeric while for the evolving transformation system the learning process combines both structural and numeric components.

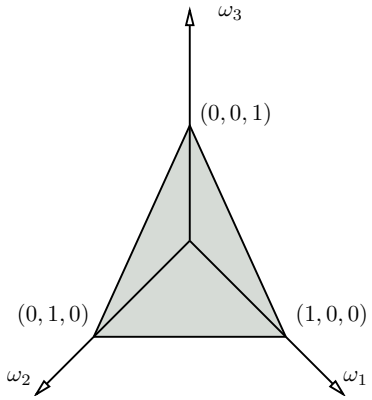


Figure 4.3: A two-dimensional unit simplex in the three dimensional parametric space Ω.

that generates the most distinctive metric configuration for the class within the global training set. The space of all the possible parameter sets Ω can be described by the $(m-1)$-dimensional unit simplex in $\mathbb{R}^m$, given by (Abela, 2001; Goldfarb, 1990; Goldfarb *et al.*, 1996)

$$\Omega = \left\{ \omega = \left(\omega^1, \omega^2, \ldots, \omega^m\right) | \, \omega^i \geq 0, \quad \sum_{i=1}^{m} \omega^i = 1 \right\}.$$

Figure 4.3 shows a two dimensional unit simplex in the three dimensional parametric space Ω. In order the search for an optimal solution in the parameter space Ω, one can use the two nonlinear function optimisation techniques briefly outlined in Section 4.3.1.2. The exhaustive search for the optimal parameter set on the entire simplex is usually computationally very expensive. Therefore, in practice (as we shall see in Section 4.3.2), it is often sufficient to evaluate f at the m vertices of the simplex (and possibly the m

corresponding midpoints on the edges connecting the vertices).

4.3.1.2 Simplex Method for Functional Optimisation

While there are many techniques for optimising nonlinear functions (Jacoby *et al.*, 1972), the most common strategy used for unconstrained optimisation and the quickest to converge is the simplex method.[4]. Simplex search, formulated by Nelder and Mead (1965), is a direct search method in that search is guided by evaluating the target function with various combinations of values of the free parameters Ω in the function. The derivative information is not used. The Nelder-Mead simplex method moves a geometric shape, called a *simplex*, through the search space using a set of well-defined transformation operations called reflection, expansion and contraction (Walters *et al.*, 1991). Each operation moves one or more of the vertices of the simplex so as to relocate the volume of the simplex closer to the optimal value of the target optimum. No general convergence properties of the simplex search strategy have been proved, but some limited proofs of convergence are known (Lagarias *et al.*, 1998; McKinnon, 1998). Some remedies for detection of non-optimality were proposed by Kelley (Kelley, 1999). An alternative to the Nelder-Mead method is to use the generally slower, but more robust, Powell method. The Powell technique is a direction set method which employs Brent one-dimensional search in each direction (Press *et al.*, 1986). Similar to the Nelder and Mead approach, choice of successive directions by the Powell technique does not require the calculation of a gradient.

All of the above mentioned numeric optimisation techniques have been previously explored by the author in his master's thesis in the context of statistical language modelling (Gutkin, 2000).

4.3.2 Evolving Transformation System

Goldfarb (1990) observed that when the set O of transformations is not sufficient to achieve a complete separation of C^+ and C^-, the structure of the model could be altered to allow for the modification of the set O. This is achieved by adding some new transformation operations. Each new transformation represents a composition of several initial operations. Modification of the transformation set leads to a new transformation system. Addition of the operations has the effect of changing the geometry of the distributions of object classes in the corresponding environment: new shorter transition

[4] The simplex method for functional optimisation and the Dantzig simplex method for linear programming (Lange, 1968, Chapter 10) both use the geometrical concept of a simplex. The two algorithms are unrelated, however. The comparison of simplex direct search method to the various gradient descent-based methods used in neural networks (such as back-propagation algorithm) is outside the scope of this thesis.

paths are generated between some pairs of objects in the structural object set S.

This leads to the central concept of the ETS$_0$ model — the mathematical structure called the *evolving transformation system*, which is constructed as a sequence of transformation systems (Goldfarb, 1990, 1992; Goldfarb *et al.*, 1996; Goldfarb and Nigam, 1994):

Definition 4.10 (Evolving Transformation System)**.** An *Evolving Transformation System* (ETS) is a sequence of transformation systems (given in Definition 4.8) that share a set S of structural objects. Each transformation system is given by

$$T_i = \left(S, O_i, D_i\right),$$

where each set of operations O_i, except O_0, is obtained from O_{i-1} by adding to it one of several operations that are constructed from the operations in O_{i-1} with the help of a small fixed set R of *composition rules*. Each rule $r \in R$ specifies how to (systematically) construct the corresponding new operation from its operands. □

From the above definition it follows that at the stage t of the structural component of the learning process

$$O_0 \subseteq O_1 \subseteq \ldots \subseteq O_t\,.$$

Furthermore, for all steps $0 \leq i \leq t-1$

$$\forall s_1, s_2 \in S, \quad \forall \Delta_{\omega_1} \in D_i \quad \exists \Delta_{\omega_2} \in D_{i+1}\colon \quad \Delta_{\omega_1}(s_1, s_2) \leq \Delta_{\omega_2}(s_1, s_2)\,,$$

where $\omega_1 \in \Omega_i$, $\omega_2 \in \Omega_{i+1}$. The dimensions of the simplex Ω_i are smaller than the dimensions of simplex Ω_{i+1}, in other words

$$\Omega_0 \subset \Omega_1 \subset \ldots \subset \Omega_t\,.$$

Each stage i, therefore induces a new topology represented by Ω_i (addition of a new transformation resulting in the growth of the simplex is depicted in Figure 4.4).

In other words, the numeric optimisation process described in the previous section by equation (4.2), becomes an inner loop within the general inductive learning process (outlined next in Section 4.3.3) that proceeds by constructing a sequence of transformations $\left\{O_i\right\}$ in such a way that, for each sequentially obtained transformation system $T_i = (S, O_i, D_i)$, the inter-distances in C^+ expressed by α_{D_i} shrink to zero while the corresponding distance β_{D_i} between C^+ and C^- remains non-zero. At each step, a more optimal family of distances is *induced* (both numerically and structurally) by a modified set of transformations. Pictorially, this process can be represented in exactly the same way as for the pure numeric optimisation, shown in Figure 4.2.

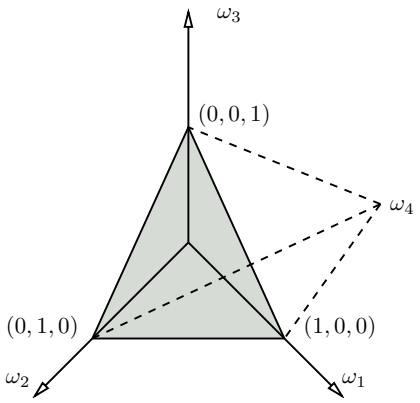


Figure 4.4: A two-dimensional unit simplex in the three dimensional parametric space Ω_i (from Figure 4.3) is grown by one dimension, turning it into a three dimensional simplex in a four dimensional parametric space Ω_{i+1}.

4.3.3 Learning in Evolving Transformation System

In general, the learning algorithm is supplied three sets. The first two are the set S of structural objects and the set of *trivial* structural transformations O_0 which operate on these objects. By trivial structural transformations we essentially mean the transformations expressing some *a priori* knowledge of the structural objects in the domain S. For the class of strings over a finite alphabet, the set of trivial structural transformations O_0 may consist of single-character edit operations (Section 4.2.1). In addition, we are given some default numeric weight vector ω_0 (which, together with O_0 induces the parametric family of distance functions D_0 over a unit simplex) whose components correspond to the transformation operations in O_0. These three parameters define the initial transformation system T_0.

In the previous section we mentioned that the learning process within ETS$_0$ essentially employs two sub-processes — the *optimisation* sub-process and the *transformation construction* sub-process. The transformation construction sub-process, which can be seen as the *structural* component, is nested within the optimisation sub-process, which is seen as the *numeric* component. Both the sub-processes can be implemented as loops. At each iteration i through the optimisation loop, the goal of the learning is to find a new transformation system T_i, which is more optimal (in terms of the optimisation of f from equation (4.2) over the unit simplex) than the current one.

At the beginning of each iteration, the optimisation loop attempts to locate an optimal weighting scheme ω_i for the current set of transformations. The transformation construction sub-process is invoked next. It iteratively constructs new candidate sets of transformations O_i^j out of the current ones O_{i-1} using the the composition rules R (see Definition 4.10). Out of these new sets of transformations, the most optimal one (in

terms of f) is chosen as the current set O_i. The algorithm then continues to the next iteration. The learning stops when the overall optimisation criterion is satisfied, i.e. when f exceeds some supplied threshold τ. At the end of the learning, the resulting transformation system $\hat{T}$ represents the most optimal configuration (the interpretation of the end-result of the learning is given in Section 4.3.4). The above basic architecture is schematically represented in Figure 4.5.

```
LEARN(S, O_0, D_0, τ)
 1  T_0 ← (S, O_0, D_0)
 2  Choose a default weight scheme ω_0 ∈ D_0.
 3  i ← 0
 4  while f(ω_i) < τ do
 5          Optimise f(ω_i) over unit simplex Ω_i obtaining ω_{i+1}.
 6          D_{i+1} ← {ω_{i+1}}.
 7          for all possible rules r ∈ R do
 8                  Construct candidate sets.
 9                  Find the most optimal (in terms of f(ω_{i+1})) set Ô_i^r.
10          O_{i+1} ← Ô_i^r
11          T_{i+1} ← (S, O_{i+1}, D_{i+1})
12          i ← i + 1
13  T̂ ← T_i
14  return T̂
```

Figure 4.5: The general architecture for learning within ETS$_0$ (Goldfarb and Nigam, 1994).

In the discussion in Section 4.3.1.1, the objective of optimisation has been specified as the maximisation, given by equation (4.2), of the following function:

$$f(\omega) \ = \frac{\beta(\omega)}{\epsilon + \alpha(\omega)}\,, \tag{4.3}$$

where $\beta(\omega)$ and $\alpha(\omega)$ are the measures of within-class proximity of C^+ and inter-class separation between C^+ and C^-, respectively. These measures can be defined in several possible ways. In this work, we essentially follow the recommendations of Goldfarb (1990; 1992) and Abela (2001) and use the following measures:

Let C^+ be the set of n positive instances and C^- be the set of m negative instances of a concept C to be learnt. The set of structural objects S is thus represented as $C^+ \cup C^-$. The measure of within-class proximity $\alpha(\omega)$ for C^+ is given by the *average within-class distance* computed over all possible pairs (s_i, s_j) in C^+ as (Abela, 2001;

Goldfarb, 1990):

$$\alpha(\omega) = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \Delta_\omega(s_i, s_j)\,, \quad \text{where} \quad (s_i, s_j) \in C^+\,. \tag{4.4}$$

The measure of separability $\beta(\omega)$ between the two classes C^+ and C^- is computed as *average interclass distance* over all pairs of objects (s_i, s_j) as (Abela, 2001; Goldfarb, 1990):

$$\beta(\omega) = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} \Delta_\omega(s_i, s_j)\,, \quad \text{where} \quad (s_i, s_j)\colon s_i \in C^+\,,\ s_j \in C^-\,. \tag{4.5}$$

Alternatively, the separability measure can be computed as the minimum distance taken over all the pairs of objects in the two sets (Abela, 2001; Goldfarb, 1990):

$$\beta(\omega) = \min\bigl\{\Delta_\omega(s_i, s_j) \,|\, s_i \in C^+, s_j \in C^-\bigr\}\,. \tag{4.6}$$

In his grammatical inference application called *Valetta*, Abela (2001) proposed a variation of the above scheme that makes use of a smaller subset $\hat{C}^-$ of the objects in C^- instead of C^- in the equations above. Abela (2001) argues that this is done in order to reduce the amount of noise in the negative training set. The subset $\hat{C}^-$ consists of 10% of the *closest* objects in C^-, which helps to remove noisy outliers, which otherwise may corrupt the quality of the measure given in equation (4.6). It should be noted that for pattern recognition applications, like our phonological representation, training set pruning is usually accomplished with the help of the clustering pre-processing step (see Chapter 2). Therefore we assume that the set C^- has already been reduced appropriately.

Both measures $\alpha(\omega)$ and $\beta(\omega)$ require a polynomial number of distance computations. More precisely, to compute $\alpha(\omega)$ one needs $n(n-1)$ distance computations, where n is the size of C^+. The calculation of $\beta(\omega)$ takes nm computations, where m is the size of C^+.

In Section 4.3.1 we mentioned that the phonological transformation system (constructed from the distinctive feature-based phonological representation) consists of N stream-specific transformation systems, where N is the number of streams (strings over some finite quantisation alphabet), as shown in example Figure 4.1. We mentioned that the numeric (weight) optimisation process in the phonological transformation system reduces to N independent optimisations (the stream independence assumption discussed in Section 2.4.1). For the inductive discovery of the optimal phonological transformation system, the above numeric optimisation can be extended to also handle the transformation construction sub-process. We conduct the learning process in the phonological

evolving transformation system by performing by N separate optimisations, one per-stream. This raises the following two important points regarding the learning process:

- The learning is conducted on the set of phonological templates $\mathbb{P}$, which can be decomposed into N sets of distinctive feature streams $\mathbb{S}_i$, $0 \leq i \leq N$. The goal of each of the N sub-optimisations is to obtain an optimal stream-specific transformation system $\hat{T}_i^{\mathbb{S}}$, where the optimisation criterion is template, rather than stream, specific. In other words, within the basic architecture of the learning algorithm from Figure 4.5, f is calculated on the candidate sets of templates, rather than streams. This ensures the learning satisfies the optimisation criteria in the global space $\mathbb{P}$, while making use of the independence assumption to simplify learning in the structural sub-process.
- Since from a structural point of view, learning within the phonological template space reduces to learning within the stream-specific (i.e. string) space, we need to explicitly specify the learning algorithm operating on the String Transformation Systems.

Below, we next address the above issues by describing both the numeric and the structural learning components in more detail.

4.3.3.1 Inductive Learning in String Transformation System

In this section a learning algorithm for a string transformation system is presented. The algorithm is a realisation of the formal discussion of learning in ETS$_0$ presented above. It is a modification of the grammatical inference algorithm initially proposed for ETS$_0$ by Goldfarb, Santoso and Nigam (1996; 1994).

Let C^+ and C^- be the respective positive and negative training sets consisting of strings over some finite alphabet Σ. In our representation, these sets represent distinctive feature streams. From Example 4.1, for instance, the set C^+ may represent all the examples of [sonorant] stream in all the instances of the phonemic class [p], while the negative samples C^- represent all the instances of [sonorant] stream in the phonemic class [b].

4.3.3.1.1 (1) Initialisation:

- We are given the set of complete transformation operations O_0 over Σ, which consists of single character edit operations. Let m_0 ($m_0 = |\Sigma|$) be the number of transformations in O_0.

- We are given the set of composition rules R, consisting of m_0 single character rules. This set allows, at any given step t of an algorithm, to construct new transformations from the existing transformations by concatenating a single character r from R to the left and right-hand sides of the transformations in O_t.
- We are also given the weight vector $\omega_0 \in \Omega_0$ corresponding to the set O_0. This vector is normalised, i.e. $\omega_0 = \{1/m_0, 1/m_0, \ldots, 1/m_0\}$.
- The algorithm used throughout the learning for computing the distances between the various instances of the streams is the Generalised Wagner-Fisher technique described in Section 4.2.2.2.

The algorithm consists of two parts, the outer and inner loops (as shown in Figure 4.5). In general, at each step of an outer loop, an inner loop yielding a new transformation system is executed. The structure of outer loop, corresponding to step (4) of an algorithm in Figure 4.5, is described below:

4.3.3.1.2 (2) Outer Loop (Simplex Optimisation): At step t, the current configuration is represented by the transformation system $T_t = (S, O_t, \Omega_t)$. The optimisation criterion at this step is represented by the function

$$f_t(\omega) = \frac{\beta_{\Delta_t}(\omega)}{\epsilon + \alpha_{\Delta_t}(\omega)},$$

where $\Delta_t(\omega)$ is induced by the current set O_t of transformations.

(2.1) Compute the value of f_t for the "centre" of the simplex Ω_t defined by

$$\omega_t = \{1/m, 1/m, \ldots, 1/m, 0, 0, \ldots, 0\}.$$

The first m_0 values of the weight vector correspond to trivial single symbol transformations and are equal to $1/m$. The last $m_t - m_0$ values correspond to newly discovered transformations and are set to 0 [5].

(2.2) If the value of f_t reached the maximum τ or if it cannot be increased further (compared to the previous value of f_{t-1}), *stop* and return T_t as the result. Otherwise, proceed to the next step.

(2.3) Perform structural optimisation in an inner loop, which is specified in step (3). This step is called the *optimal transformation construction*. Optimal transformation construction results in a new transformation system T_{t+1} which has $m_{t+1} - m_t$ new transformation operations. These new transformations are supposed to improve f_t. Finally, update t to $t + 1$ and return to the step (2.1).

[5] Hence, the point ω_t does not quite correspond to the centre of the simplex.

It is an inner loop, discussed next, which ensures that the new pseudo-metric space learnt at stage t and the corresponding transformation system improve the current value of f_t:

4.3.3.1.3 (3) Optimal Transformation Construction: The current iteration is given by t and the current set of transformations is O_t.

(3.1) Compute the values of f_t at those vertices of weight simplex Ω_t that correspond to the m_0 trivial one symbol transformations. In other words, the set of vertices, for which the values of f_t are computed, consists of the following m_0 vectors ω_i of dimension m_t

$$\omega_i = (\omega_i^1, \omega_i^2, \ldots, \omega_i^{m_0}, 0, 0, \ldots, 0)\,, \quad 1 \leq i \leq m_0\,,$$

where $\omega_i^k = 1$ for $k = i$ and $\omega_i^k = 0$ for $k \neq i$. Obviously, each vector ω_i corresponds to a trivial transformation o_i^0 in the initial transformation set O_0.

(3.2) Promote to the next stage the set $O_0^* \in O_0$ of all transformations for which the value of $f_t(\omega_i)$ is *maximal* (the construction of weight vectors ω_i is given above).

(3.3) Form the set U_2 of all possible two-symbol candidate transformations from the one-symbol transformations $o_i^0 \in O_0^*$ promoted in the previous step. This is achieved by employing left and right concatenation of the single character rules r from the set R.

This step spawns the *search for candidate expanded operations*, which starts in the next step. Initialise the current iteration counter l to $l = 2$ and let U_l be the initial set of non-trivial candidate transformations of length 2, constructed above. For each candidate $u_l^i \in U_l$, an *ancestor* $\mathrm{anc}(u_l^i)$ is defined as the corresponding candidate u_{l-1}^j of length $l-1$ out of which u_l^i was constructed.

(3.4) For each of the candidate transformations u_l^i in F_l, each of length l, check whether this transformation is present as a substring in C^+. If it is present, add it to the set U_l' of candidate transformations which passed the matching test and proceed to the next step. *Otherwise*, if $l > 2$, add the ancestor $\mathrm{anc}(u_l^i)$ of u_l^i to the set of current operations O_t and *finish* the stage (3) (Optimal Transformation Construction).

(3.5) Let K be the number of the transformations promoted in the previous step. Add each of the K promoted transformations u_l^i in U_l' ($K = |U_l'|$), one at a time, to the current set of transformations O_t, obtaining K augmented sets O_t^i. All these sets are of the size $m_t + 1$. For each of the K sets O_t^i, compute the value of f_{t+1}

at the "centre" ω^i_{t+1} of the corresponding candidate simplex Ω^i_{t+1}, which is given by the $(m_t + 1)$-dimensional vector whose first m_0 values are equal to $1/m_0$, and the rest are zero.

(3.6) Promote all the transformations u^i_l for which the corresponding values of $f_{t+1}(\omega^i_{t+1})$, computed above, are *minimum*.

(3.7) On the basis of the l-symbol transformations promoted in the previous step, form the set U_{l+1} of candidate $l+1$-symbol transformations, employing the composition rules from R (left and right concatenation). Set l to $l+1$ and return to step (3.4).

4.3.4 Inductive Class Representation

In view of the above, the goal of the inductive learning problem stated in Definition 4.1 has been specified more concretely in (Goldfarb, 1990, 1992; Goldfarb and Nigam, 1994):

Definition 4.11 (Inductive Class Representation)**.** The *inductive class representation* or *inductive generalisation* is defined as a 3-tuple

$$\Pi = \left(\hat{C}^+, \hat{O}, \hat{\Omega}\right), \tag{4.7}$$

where $\hat{C}^+$ is a subset of C^+, $\hat{O}$ is the final set of transformations at the end of the learning process, and $\hat{\Omega} \subseteq \Omega$ is a set of optimal weight vectors $\{\hat{\omega}\}$ corresponding to the final transformation system. □

The elements of $\hat{C}^+$ act as reference patterns for defining the class. Consequently, a new input pattern is always compared with these reference patterns using the parametric family of distance functions $\{\Delta_{\hat{\omega}}\}$ induced by $\hat{\Omega}$. The set $\hat{O}$ is necessary since the concept of a distance can be defined properly only in terms of these operations.

Example 4.2. Figure 4.6 shows the non-trivial stream-specific transformations discovered during the learning process for the two-class phone problem, treated in Examples 2.3 (Figure 2.5) and 4.1 (Figure 4.1).

These operations (the corresponding optimal sets of weights $\hat{\Omega}_{/p/}$ and $\hat{\Omega}_{/b/}$ are not shown) together with the trivial one-symbol transformations form the optimal set of transformations for each class. Together with the corresponding sets of reference objects $\hat{C}^+_{/p/}$ and $\hat{C}^+_{/b/}$, the three-tuples

$$\Pi_{/p/} = \left(\hat{C}^+_{/p/}, \hat{O}_{/p/}, \hat{\Omega}_{/p/}\right) \quad \text{and} \quad \Pi_{/b/} = \left(\hat{C}^+_{/b/}, \hat{O}_{/b/}, \hat{\Omega}_{/b/}\right)$$

provide inductive class representations for the two classes in question.

It is important to mention that each set of reference objects ($\hat{C}^+_{/p/}$ and $\hat{C}^+_{/b/}$) consists of one template only. This template is arbitrarily chosen from the corresponding

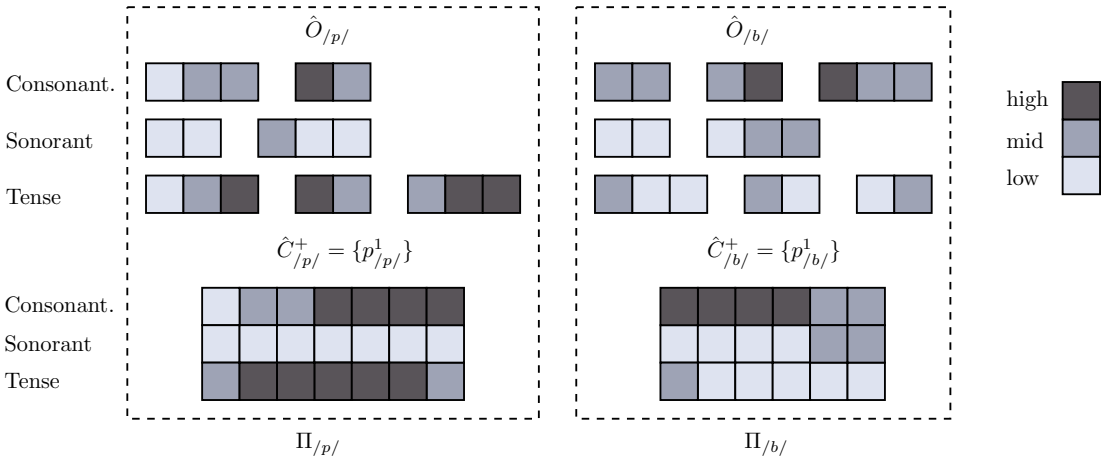


Figure 4.6: Discovered per-stream feature transformations ($\hat{O}_{/p/}$ and $\hat{O}_{/b/}$) (corresponding to the representation in Figure 2.5 on p. 58 and Figure 4.1 on p. 133) and the resulting class representations $\Pi_{/p/}$ and $\Pi_{/b/}$.

set. This is because, in this particular example, the discovered transformations induce perfect separation between the two classes. The within-class distance for $C^+_{/p/}$ (which contains two templates) induced by $\hat{O}_{/p/}$ is zero (while the corresponding distance to $C^+_{/b/}$ is non-zero). Hence, we can choose a single object as a compact representation of $C^+_{/p/}$, without loss of generality. The situation is analogous with $C^+_{/b/}$.

Hence, not all the transformations found in the optimal transformation set $\hat{O}$ are necessarily found in the corresponding set of reference patterns. For instance, the first three-character transformation for the [tense] stream, shown in Figure 4.6, which corresponds to $\hat{O}_{/p/}$, is not found in the corresponding reference set $\hat{C}^+_{/p/}$ containing the pattern $p^1_{/p/}$. It is, however, part of the training data and is found in pattern $p^2_{/p/}$ (Figure 4.1 on p. 133). Therefore, we conclude this example by the following important observation: transformations comprising an optimal transformation set are fragments of training examples which are *always* found in the original training set. They, however, are not always found in the set of reference objects chosen as class representation, because the size of this set is usually significantly smaller. ▷

The inductive representation considered above is meaningful, in the sense that it is able to capture certain consonantal properties of the phonemes [p] and [b]. For example, from this toy representation we can learn the main difference (within the postulated three-stream representation) between the two classes, namely different behaviour of the [tense] feature. In general, tense sounds are produced with a deliberate, accurate, maximally distinct gesture that involves considerable muscular effort; non-tense sounds are produced rapidly and somewhat indistinctly (Giegerich, 1992). In Figure 4.6, transformations corresponding to the [tense] stream capture the fact that within the available

examples of [p], this feature is either *high* or in the process of gradually changing (fluctuating) around the *high* values, whereas for the examples of [b], the process is opposite. This coincides with the assumption of phonological contrast between [p] and [b] phones within the SPE feature system (Chomsky and Halle, 1968). In addition, the transformations capture certain asynchronies in the process of sound changes. The first transformation corresponding to the [consonantal] stream for the class [p] indicates the change from *low* to *high* which most probably means that one (or both) of the examples were derived from the context in which they were preceded by a vowel or consonantal sounds from [w] or [j] classes.

In Chapter 2 we mentioned that each phonological template p belonging to the pseudo-metric space $\mathbb{P}$ consists of N distinctive feature streams. In our representation, we use N equal to 25 for representing five multivalued feature streams (Section 2.4.1). In the discussion of the learning algorithm for ETS$_0$ we also mentioned that for each of the classes of the phonemes, the learning reduces to N independent optimisations of the per-stream transformation systems. Each particular stream p_i^j $(1 \leq i \leq N)$ from a class C^j is optimised against the streams of the same type p_i^k belonging to all other classes $(k \neq j)$. Thus, for each of the classes C^j in the domain, the end result of the learning process essentially consists of N per-stream inductive class representations Π_i^j $(1 \leq i \leq N)$, which are the structures from Definition 4.11. Each stream-specific inductive class structure Π_i^j induces its own metric $\Delta_{\Pi_i^j}$. Hence, we can define the *phonological inductive class structure* Π^j for a class C^j as the collection of N stream-specific structures

$$\Pi^j = \left\{\Pi_1^j, \Pi_2^j, \ldots, \Pi_N^j\right\}.$$

For the symbolic space of phonological templates, the distance between any prototype template p^j in Π^j, which defines the class C^j, and an unknown template p is given as a linear combination of the individual metrics induced by different constituent streams. In other words,

$$\Delta^j(p^j, p) = \sum_{i=1}^{N} \Delta_i^j(p_i^j, p_i)\,.$$

4.4 Experiments and Discussion

In this section we present the experimental results of a phoneme classification task on the structural data corresponding to the phonological templates, derived from the TIMIT database of read speech. The experimental setup mirrors the one described in detail in Section 2.6 of Chapter 2, where we also mention the construction of the phonemic templates from real speech and reduction of the size of the training set.

Similar to the strategy adopted for the experiments in pseudo-Euclidean spaces (Chapter 3), we split the experiments into two parts. In order to get a better idea of the performance of the learning and classification within ETS$_0$, we first focus on a smaller 3-class task, where the classes are *a priori* reasonably separable. Since this task is small, it allows us to analyse the performance of the learning algorithm in more detail. Experiments on a 3-class problem are described in Section 4.4.1. In order to compare the performance of the classifiers in the original symbolic space (Section 2.6) and the new pseudo-metric spaces induced by ETS$_0$ on a standard full-class TIMIT task, in Section 4.4.2 we describe the results of experiments involving 39 classes of phonemes. We also compare the performance of the system constructed using ETS$_0$ with the performance of the baseline symbolic algorithms from Chapter 2 and the vector space algorithms from Chapter 3, obtained on both tasks.

For both the three class and the full tasks, the learning essentially consists of one-against-all optimisation of each of the classes in the training set against all others. The structure of the resulting class-specific metrics, obtained in ETS$_0$ space, is given in Section 4.3.4. During the classification stage, we use the extension of the k-NN AESA search (described in Chapter 2) that makes use of the metrics obtained at the end of the learning in ETS$_0$. The baseline corresponds to the performance of the k-NN AESA search in a "rigid" pseudo-metric space described in Chapter 2. The best results (on the full 39-class task) were obtained in the following optimal settings:

- The set P corresponds to the set of phonological templates derived from the TIMIT data using a symbolic quantisation level of 10. The symbolic corpus corresponding to this quantisation level consists of 124,962 templates in the training set and 42,540 templates in the test set.
- The similarity function D_P, operating on the phonological templates from the set P, corresponds to the weighted Levenshtein distance.
- The clustering technique is the k-medians, employing phonological set median (rather than generalised median) and duration-based initialisation.

This is the same baseline used by the experiments in pseudo-Euclidean vector spaces described in Section 3.5. The stopping criterion τ employed by the ETS$_0$ learning algorithms (Figure 4.5) was chosen to be 10^{-8} throughout the experiments described below.

4.4.1 Three-class Problem

Similar to the experimental setup described in Section 3.5.1, the first set of experiments focuses on classification of three classes of phonemes from three different phonological

categories which are *a priori* known to be reasonably separable (Ladefoged, 2001). The three classes under investigation consist of one vowel [aw] (low back round) and two consonants [b] (voiced bilabial stop) and [z] (voiced alveolar fricative). The original training set for these three classes consists of 6,629 unique symbolic phonological templates. The entire test set for the three classes of phones, consisting of 2,423 unique phonological templates, was used in this experiment.

First, the baseline for the three-class problem was created by breaking down the training set into a smaller subset using the best performing dimensionality reduction technique. This corresponds to k-medians, employing phonological set median and duration-based initialisation. Six training datasets were created, for which the number of examples per class $|P|$ is 5, 10, 15, 30, 50 and 100 (shown in the first column of Table 4.1). In order to obtain the baseline classification results, we conducted classification experiments on the full test set of 2,423 templates, using each of the training datasets above. The classification rule we used corresponds to the best performing classification technique from Chapter 2: the k-NN AESA search (employing weighted Levenshtein distance with normalised weights) with the size of the k-best list of 1. The performance of the baseline models (in terms of classification error) is shown in the second column of Table 4.1.

In order to compare the performance of the baseline algorithms with the performance of the learning algorithms we use in ETS$_0$ framework (Section 4.3.3), additional experiments were conducted for each of the six datasets. In Section 4.3.1.1 we mentioned that the objective of the learning is specified as the maximisation (equation (4.2)) of the function f given in equation (4.3). Function f is defined as the ratio between the inter-class separation measure β and the the within-class proximity measure α plus some small constant ϵ to prevent the overflow. We also mentioned two potential ways of computing the separability measure β: as an *average inter-class distance* (given in equation (4.5)) or as the *minimum inter-class distance* (given in equation (4.6)). Hence, it is possible to choose among the two functions to optimise: f^r (where the superscript r stands for *regular*) and f^m (where m stands for *minimum*), corresponding to the average and minimum inter-class distances respectively.

The results (in terms of classification error) for the models trained using these criteria are shown in columns three and four of Table 4.1. The algorithms are denoted $L^r_{\mathrm{ETS}_0}$ and $L^m_{\mathrm{ETS}_0}$, respectively.

Performance of the k-NN AESA search in the symbolic space constructed using the learning algorithm $L^r_{\mathrm{ETS}_0}$, employing the optimisation criterion f^r given by

$$f^r(\omega) \ = \frac{\beta^r(\omega)}{\epsilon + \alpha(\omega)} \, ,$$

$\|P\|$	k-NN AESA	$L^r_{\mathrm{ETS}_0}$	$L^m_{\mathrm{ETS}_0}$
5	1.5	1.0	2.1
10	1.4	1.6	2.0
15	1.3	1.7	1.9
30	1.0	1.0	**1.3**
50	1.0	**0.8**	1.8
100	**0.9**	0.9	1.4

Table 4.1: Three-class task: Performance of the k-NN AESA symbolic search in the spaces constructed by ETS$_0$ learning algorithms ($L^r_{\mathsf{ETS}_0}$ and $L^m_{\mathsf{ETS}_0}$) and the "rigid" metric space baseline from Chapter 2. Best classification errors (%) are shown in bold.

is analysed first. It performs the same or better than the baseline on four out of six datasets from Table 4.1. In particular, using this learning strategy we obtained the best result of 0.8% classification error among all the experiments conducted with *symbolic* models on the three class task. When compared to the best results obtained on the same task in the pseudo-Euclidean vector space, the k-NN AESA search in the symbolic space obtained with $L^r_{\mathrm{ETS}_0}$ outperforms its numeric counterpart in all the pseudo-Euclidean spaces shown in Table 3.1. In the experiments described in Section 3.5.1, the best k-NN AESA classification error of 1% was obtained for the space constructed using the class-based corrected metric projection. Comparing this result to the other numeric models in pseudo-Euclidean vector space (neural networks and support vectors shown in Table 3.1), the performance of the search in the symbolic space constructed by $L^r_{\mathrm{ETS}_0}$ is consistently worse.

The k-NN AESA search in the symbolic space constructed using the learning algorithm $L^m_{\mathrm{ETS}_0}$, employing the optimisation criterion f^m given by

$$f^m(\omega) \ = \frac{\beta^m(\omega)}{\epsilon + \alpha(\omega)}\,,$$

appears to perform worse than both the baseline model and the model employing the $L^r_{\mathrm{ETS}_0}$ learning criterion. With the $L^m_{\mathrm{ETS}_0}$ model, the best obtained classification error is 1.3%. The model outperforms the k-NN AESA search in the pseudo-Euclidean spaces constructed using the regular basis selection technique (denoted kNN^R_R and kNN^R_C in Table 3.1). Overall, however, the performance of this model is not satisfactory because it performs consistently worse than other symbolic models.

The reason why $L^r_{\mathrm{ETS}_0}$ learning, based on the average inter-class distance, outperforms the $L^m_{\mathrm{ETS}_0}$, based on the minimum distance, becomes clearer if we investigate the values of the corresponding functions f^r and f^m during the optimisation process.

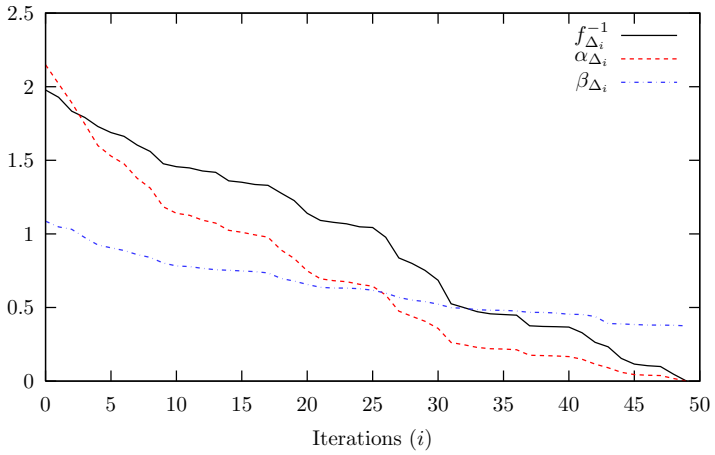


Figure 4.7: The measures of the average within-class distance, the average inter-class distance and their ratio, shown during the $L^r_{\mathsf{ETS}_0}$ learning in one of the streams of phoneme [aw] ($f_{\Delta_i} = f^r_{\Delta_i}$).

For $L^r_{\mathrm{ETS}_0}$, the measures of the average within-class distance, the average inter-class distance and the inverse of the corresponding f^r are shown in Figure 4.7. These curves correspond to the learning process in one of the streams of [aw]. As can be seen from Figure 4.7, the learning converges in 49 steps, obtaining $\alpha = 0$, which is the most optimal within-class configuration. In this particular case, all the streams of this type belonging to [aw] can be *generated* from any other stream using the discovered non-trivial transformations (with zero weights) only. This is because single character transformations, whose weights are non-zero, do not contribute to the overall within-class distance measure since non-trivial transformations are preferred[6]. The average inter-class distance β decays more slowly and stays non-zero when the learning process completes. This is an indication that the original expectations of *simultaneous* minimisation of within-class and maximisation of between-class distances were rather premature. The discovered transformations do not increase the separation between the classes. We do not think that this issue is very problematic because our minimal expectation that the between-class distance should be non-zero is fulfilled.

For $L^m_{\mathrm{ETS}_0}$, the measures of the average within-class distance, the minimum inter-class distance and the inverse of the corresponding f^m are shown in the two subfigures of Figure 4.8. Two different scenarios encountered during the learning of two distinct streams of [b] are shown in Figures 4.8a an 4.8b. The first scenario from Figure 4.8a corresponds to the expected functioning of the learning process that converges in 10 steps. The overall behaviour of the system is similar to that corresponding to the

[6] An interesting interpretation of this fact (due to Abela, 2001) is that the single character transformations represent uncertainty (or "noise"). By bringing the within-class distance to zero, we essentially remove the noise from the class.

case of $L^r_{\mathrm{ETS}_0}$, with both the average within-class distance α and f^{-1} decaying until the convergence criterion is met. The curve corresponding to the minimal inter-class distance β reaches a steady state before the system converges.

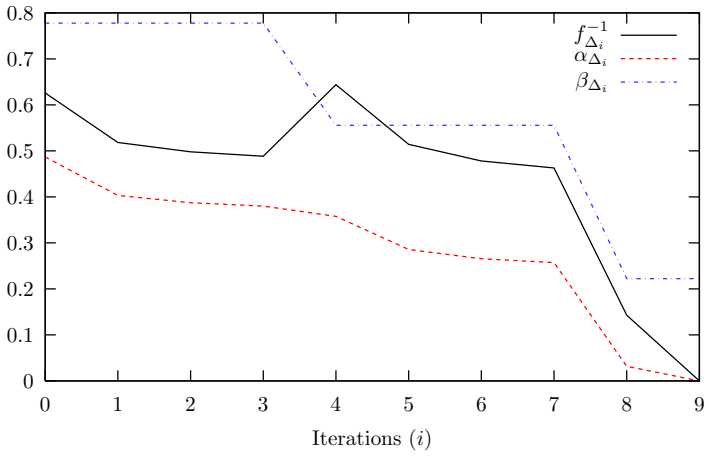
The second scenario (shown in Figure 4.8b) corresponds to the problematic case when there exists some overlap between the positive and negative datasets. Despite the fact that the classes of phonemes are reasonably separable, the classes of streams need not be. In such a case, the minimal inter-class distance remains 0 throughout the learning, which affects the overall optimisation criterion f^m (the inverse of f^m is shown in Figure 4.8b as a $\frac{1}{8}$ fraction of the decimal log scale). Despite the fact the learning process converges in 13 steps, obtaining an optimal within-class configuration, it does not achieve an adequate separation between the given class and the rest simply because of the presence of overlapping outliers which are shared between the two classes and "corrupt" the between-class separability measure β. Essentially, this case can be seen as a one-class learning scenario, where the set of negative samples is empty. We hypothesise that such cases cause the overall quality of the $L^m_{\mathrm{ETS}_0}$ learning to deteriorate and are the cause of the inferior performance.

The average number of transformations (excluding the ones of length one) per class ($\overline{N}_{\mathrm{ETS}_0}$) and the average transformation length ($\overline{F}_{\mathrm{ETS}_0}$) discovered by the $L^r_{\mathrm{ETS}_0}$ and $L^m_{\mathrm{ETS}_0}$ learning algorithms are shown in Table 4.2. Both learning strategies prefer shorter transformations (given by $\overline{F}^r_{\mathrm{ETS}_0}$ and $\overline{F}^m_{\mathrm{ETS}_0}$). This is because of the nature of the composition rules which construct candidate operations by left and right concatenation of single symbols. It also appears that on average, the $L^m_{\mathrm{ETS}_0}$ strategy discovers more transformations per class, which are also slightly more compact than the transformations discovered by $L^r_{\mathrm{ETS}_0}$. Both the $\overline{N}^r_{\mathrm{ETS}_0}$ and $\overline{N}^m_{\mathrm{ETS}_0}$ refer to the average number of transformations per *class of templates*. They have to be divided by the number of streams (25) to obtain the average number of transformations discovered *per-stream* for each class ($\overline{S}^r_{\mathrm{ETS}_0}$ and $\overline{S}^m_{\mathrm{ETS}_0}$).

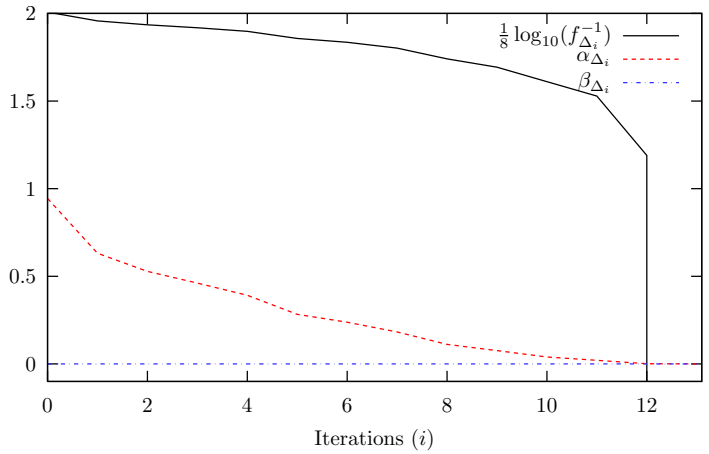
4.4.2 Full Problem

The symbolic database consists of 124,962 templates in the training set and 42,540 templates in the test set. The full-class task consists of evaluating the performance of ETS_0 models of 39 phonetic classes against 42,540 phonological templates representing the test objects. Similar to the clustering setup described in the previous section and Section 2.6, the training dataset is divided into the smaller sets of 5, 10, 15, 30, 50 and 100 objects per class.

Phoneme classification experiments were conducted on the full 39-class task using the ETS_0 models constructed by the $L^r_{\mathrm{ETS}_0}$ and $L^m_{\mathrm{ETS}_0}$ learning algorithms. The results,



(a) Normal functioning.



(b) Problematic case.

Figure 4.8: The measures of the average within-class distance, the minimum inter-class distance and their ratio, shown during the $L^m_{\mathsf{ETS}_0}$ learning in two different streams of phoneme [b]. Normal (Figure 4.8a) and problematic (Figure 4.8b) situations are shown. In Figure 4.8b, f^{-1} is shown on a log scale ($f_{\Delta_i} = f^m_{\Delta_i}$).

$\|P\|$	$\overline{N}^r_{\text{ETS}_0}$	$\overline{F}^r_{\text{ETS}_0}$	$\overline{S}^r_{\text{ETS}_0}$	$\overline{N}^m_{\text{ETS}_0}$	$\overline{F}^m_{\text{ETS}_0}$	$\overline{S}^m_{\text{ETS}_0}$
5	99	3	4	87	3	3
10	152	3	6	139	3	5
15	230	3	9	199	3	8
30	382	4	15	310	3	12
50	523	5	21	498	4	20
100	684	5	27	643	4	26

Table 4.2: Three-class task: Average number of transformations (excluding transformations of length one) per class ($\overline{N}_{\text{ETS}_0}$), the average transformation length ($\overline{F}_{\text{ETS}_0}$) and average number of transformations per stream ($\overline{S}_{\text{ETS}_0}$) discovered by the $L^r_{\text{ETS}_0}$ and $L^m_{\text{ETS}_0}$ learning algorithms. These values are training dataset-specific.

shown in Table 4.3, are then compared to the results of the symbolic baseline algorithms (Table 2.5) which we evaluated in Chapter 2.

$\|P\|$	$L^r_{\text{ETS}_0}$	$L^m_{\text{ETS}_0}$
5	50.6	48.2
10	52.8	49.9
15	53.1	**52.3**
30	**54.2**	52.1
50	52.7	52.3
100	51.9	51.4

Table 4.3: Full task: Performance of the k-NN AESA symbolic search in the spaces constructed by ETS$_0$ learning algorithms ($L^r_{\text{ETS}_0}$ and $L^m_{\text{ETS}_0}$) and the "rigid" metric space baseline from Chapter 2. Best classification accuracies (%) are shown in bold.

Similar to the three-class experiments described in the previous section, the k-NN AESA search in the symbolic space constructed using the learning algorithm $L^r_{\text{ETS}_0}$, which employs the average inter-class distance criterion, appears to outperform the search in the symbolic space constructed with $L^m_{\text{ETS}_0}$. The best result obtained with the $L^r_{\text{ETS}_0}$ model is 54.2% accuracy, compared to the 52.3% accuracy obtained with $L^m_{\text{ETS}_0}$. From this result and the result obtained for the smaller task, we can conclude that the optimisation criterion employing the average inter-distance rather than minimum inter-distance, is preferable for learning within the phonological ETS$_0$ representation

under investigation.

Both models appear to consistently outperform only one of the baseline models shown in Table 2.5, which corresponds to the setup employing the combination of weighted Levenshtein metric, mean templates and *MaxMin* clustering initialisation ($M^G_{\mathbb{P}}/D^L_{\mathbb{P}}/K^M_{\mathbb{P}}$). Furthermore, the best result of 54.2% obtained with $L^r_{\mathrm{ETS}_0}$ is better than the result of 54.12% obtained with the baseline model corresponding to the combination of weighted Levenshtein metric, median templates and *MaxMin* clustering initialisation ($M^S_{\mathbb{P}}/D^L_{\mathbb{P}}/K^M_{\mathbb{P}}$).

The other two baseline models considered in Chapter 2, consistently outperform both the $L^r_{\mathrm{ETS}_0}$ and the $L^m_{\mathrm{ETS}_0}$ models. In particular, we were unable to surpass (on any of the datasets considered above) the best performing (60.3%) symbolic baseline model which corresponds to the combination of weighted Levenshtein metric, median templates and duration-based clustering initialisation criterion. We hypothesise that the reason for this unsatisfactory performance of the $L^r_{\mathrm{ETS}_0}$ and $L^m_{\mathrm{ETS}_0}$ models on the full-class task has to do with the properties of the resulting distance function induced during the learning. In order to ascertain this, we conducted the following experiment: Given the test set of 42,540 phonological templates, we performed 4,836,564 tests of the triangle inequality. The metric we evaluated was induced by the $L^r_{\mathrm{ETS}_0}$ algorithm on the dataset corresponding to 15 templates per class. The percentage of violations of the triangle inequality in this experiment was 0.03%, which corresponds to 1,196 failed tests. Though this number is small, it nevertheless shows the *semimetric* nature of the resulting distance function. This semimetric property, in turn, influences the quality of the k-NN AESA search, which is based on the distance computations only. An additional problem which may arise is the discovery of the "wrong" transformations during the learning. This is because the same semimetric distance functions are used during the learning for the computation of class separability and proximity measures.

4.5 Summary and Potential Improvements

In this chapter we showed how to *inductively* approach the issue of structural learning within the symbolic phonological metric spaces. By the adjective "inductive" we mean the formulation of the learning problem in such a way that the goal of the learning is the discovery of the non-trivial structural transformations which make the instances of the phonemes in each of the classes similar to each other. We approached this problem using ideas from ETS$_0$ framework, which was specifically developed to address these needs. The central idea of this approach is the discovery (during the learning) of the optimal structural transformations which induce a new and more optimal metric

space, where (at least in theory) the classification should be easier since the discovered structural transformations, participating in this new metric, lead to the better class separation. The main attraction of this approach, however, is in the representational power it affords: discovering the structural (and hence fully interpretable) transformations that make the phonemes different and/or similar is linguistically more meaningful than performing a purely numeric optimisation. We believe that the emphasis on the *structural* class representation of linguistic phenomena will facilitate the development of the speech recognition field, since the recognition problem cannot be adequately approached without a meaningful representation.

We believe that ours is the first attempt in pattern recognition to address the issue of ETS$_0$ structural learning on a large (for structural models) real-world database. Previous research (Goldfarb, 1990; Goldfarb and Deshpande, 1997; Goldfarb *et al.*, 1996; Goldfarb and Nigam, 1994) was theory oriented and focused on small-scale pattern classification experiments. The only (major) previous practical application of the framework is in the field of grammatical inference (Abela, 2001), which is obviously very different from the phonological representation and classification problem we are addressing.

Therefore, the linguistic attractiveness of the ETS$_0$ framework we considered in this chapter comes at a cost. Despite the fact that we were able to show improvements in phoneme classification on a small task, the performance of the system on a full task does not match the original expectations. This can be explained by the fact that, compared to the structurally "rigid" approaches we discussed in Chapter 2, there are many more factors in play when one considers structural learning and optimisation. Below, we discuss several areas of research we expect would lead to significant improvements in the modelling power and classification of the framework considered in this chapter.

Potential Improvements

There are several ways of improving the algorithms of the ETS$_0$ framework described in this chapter:

- In the discussion in Section 4.4, we mentioned that the optimisation criteria (especially the ones incorporating the minimum inter-class distance measure) are not very robust in the presence of the noise. The learning algorithm was primarily developed for the grammatical inference problem, where the classes are some formal languages generated by distinct grammars. These problems are usually much less "noisy" than the pattern recognition ones. Hence, we may need to look for better proximity and class-separability measures α and β to use in the optimisation function f (given by equation (4.3)).

- Although the learning algorithm already incorporates some of the changes we intended to make (such as more robust stopping criteria than those specified in Abela, 2001; Goldfarb and Nigam, 1994), more work is needed in order to investigate the behaviour of the simplex optimisation. In particular, we observed several cases of convergence of f to non-stationary points (see Section 4.3.1.2). There are several possible remedies to rectify this situation (though they will make the learning process slower):
  - Evaluate the simplex at additional points (not only at the vertices). This will allow us to tune the weights better.
  - The Powell method of non-linear optimisation is more robust than the simplex method of Nelder and Mead. Hence, better estimates can be obtained with it.
  - Using either Nelder & Mead or Powell techniques, it is possible to conduct an exhaustive search for the optimal transformation weights. This might give an optimal solution, though computationally it is totally intractable.
- Better dimensionality reduction techniques will be considered. The clustering setup resulted in the number of overlaps between several classes under investigation. This is because the clustering algorithms, described in Chapter 2, did not take into the account the need for class separation.
- In the previous section we mentioned that the semimetric properties of the distance functions employing the non-trivial transformations may corrupt the quality of the learning and classification procedures. This was first observed by Abela (2001), who suggested the use of normal forms as a remedy. Briefly, given a string and a set of transformations, a *normal form* of a string is another string with all the occurrences of the supplied transformations removed from it. This can be extended to normal forms of phonological templates in a straightforward way. The calculation of a normal form, however, is a non-trivial procedure. In general, given a string and a set of transformations, there would be several normal forms corresponding to a string (one for each distinct sequence of the removed substrings) and the algorithm will have to choose the most optimal one.

  During the learning and classification processes, once the training and test sets are reduced to their normal forms using the currently discovered transformations, the induced metric is not needed anymore and one can use the regular weighted Levenshtein distance which possesses the desired metric properties. This is because the non-trivial transformations are not present in the sets of positive and negative samples.

Incorporation of this procedure into the learning algorithm described in Section 4.3.3.1 is a non-trivial task. Ideally, one would need to reduce the sets of positive and negative objects every time the new candidate transformations are found. To every candidate transformation found, there would correspond its own copy of the training and test sets. Though the actual computation of the distances between any two objects will become much faster, the overall computation complexity may increase because of the additional requirements of the reduction process.

Part II

Structural Representation Formalism

Chapter 5

Formal Articulatory Representation of Speech with ETS_2

5.1 Introduction

In the first part of this thesis we described the structural representation of speech built around the concept of distinctive phonological features. We introduced the representation in Chapter 2. In the same chapter, as well as in Chapter 3 and Chapter 4, we focused on the classification and learning techniques in various symbolic and numeric spaces corresponding to this representation. The main theoretical difficulty which we encountered is the fact that distinctive phonological features are difficult to extract directly from speech without resorting to the use of numeric models (such as neural networks) which are able to perform the non-linear mapping between the acoustics and distinctive feature "space". While such a mapping is definitely desirable for the numeric approaches to speech modelling, it is nevertheless quite problematic from the point of view of structural modelling. This is because the main attraction of the symbolic approaches is their ability to represent (and discover) the *structure* of the process being modelled. As an example, consider the multivalued feature describing the manner of articulation. We can recover its values, e.g. fricative, from the acoustic stream using some numeric model. In the process, however, we loose all the original information (present in the data) which might have given us *structural* clues as to what is a fricative. We partially addressed this issue in Chapter 4, were the ETS_0 approach was presented. The adverb "partially" refers to the fact that while we focused on the discovery of structural features, the data on which we operated was in the distinctive phonological feature space, rather than original space.

Given the above observations, the question we address in this chapter is the following: Is there a conceptually simple way of constructing a *richer* structural representation *directly* from the data?

First of all, we observe that basing a structural representation on distinctive phonological features may not be the best of options, since these units are too abstract and thus are difficult to extract from the real data directly. There is a better alternative to distinctive phonological features. This alternative is offered by the theory of articulatory phonology (Browman and Goldstein, 1992). In articulatory phonology, vocal tract action during speech production is decomposed into discrete, re-combinable atomic units (Browman and Goldstein, 1989). The central idea is that, while the observed products of articulation (articulatory and/or acoustic measurements) are continuous and context-dependent, the physiological actions which regulate the motion of the articulators are discrete and context-independent. These atomic actions, known as *gestures*, are hypothesised to combine in different ways to form the vast array of words that constitute the vocabularies of human languages (Stevens, 1989; Studdert-Kennedy and Goldstein, 2003). This combinatorial outlook on speech places it in the same context as other natural systems (for instance, combinations of gestures are similar to molecular compounds in chemistry). Compared to traditional approaches — such as distinctive phonological features — the gestural approach is more physiologically concrete and offers a compact means of representing the truly asynchronous nature of speech, allowing for better interpretations of all-pervasive complex phonological phenomena (such as assimilation).

Second, we propose a richer structural representation of speech built around the above articulatory gestures. The representation is based on the Evolving Transformation System (ETS_2) formalism, outlined by Goldfarb *et al.* (2004). Taking into account the limitations of the original version (ETS_0), ETS_2 has been specifically proposed as a radically new formal framework for the structural representation of "natural" processes. As observed by Abler (1989), these processes, which are studied in natural sciences (such as evolutionary biology, organic chemistry and physiology) share some important combinatorial properties. In the articulatory representation we propose in this chapter, the natural process we model is the physiological process of articulation, which is captured by the fundamental concepts of the ETS_2 formalism. In particular, the discrete articulatory gestures are represented by the atomic units of the ETS_2 formalism, while non-trivial combintations of these gestures are represented as formal structures encoding the "formative history" of the corresponding objects. As will become evident from the discussion in this chapter, the articulatory representation is richer, both semantically and syntactically, than its distinctive feature-based counterpart.

Third, we note that the articulatory gestures can be extracted from the data. Since our aim is to work *directly* with the data, the (unstructured numeric) measurements need to be articulatory. In practice, some of the gestures can be detected from multiple measurement sources. For example, vibration of the vocal folds can be detected from both the laryngeal pressure waveform and the corresponding acoustic recording. The purely quantitative approach to automatic derivation of gestural structures from articulatory speech data has been studied in detail by Jung (1993; 1996), who proposed using a derived numeric representation of the gestural structure both as alternative units for continuous speech recognition and as a compact representation of the acoustic waveforms. An alternative (qualitative) approach, advocating the use of automatically derived gestures as the generic qualitative units for any *structural* (e.g. hypergraph-based) representation of continuous speech, has been proposed by us in (Gutkin and King, 2005a). In this chapter, the latter approach was used for detection of the articulatory gestures in the continuous speech data and automatic derivation of gestural structures for ETS$_2$ representation. We previously reported various features of ETS$_2$ articulatory representation in (Gutkin *et al.*, 2004; Gutkin and Gay, 2005b,c).

Finally, it is important to note that in this chapter we focus on *representation*. The issue of learning within ETS$_2$ is omitted from the discussion because the learning algorithm (described in Goldfarb *et al.*, 2004, Part III) needs more work to be usable in practical pattern recognition applications (please refer to the end of this chapter, where on p. 206 more information is provided). Therefore, in this chapter we deal with the classification of the class structures which are *a priori* postulated based on the linguistic evidence. The implementation of ETS$_2$ algorithm is available and was used by us to perform some learning experiments, which largely confirmed our hypotheses with regard to phonemic class structures. However, several problematic issues with the algorithm have prevented us from presenting the algorithm and describing those experiments here or in a separate chapter.

Overview of the chapter

The core elements of the ETS$_2$ formalism are introduced in Section 5.2. The ETS$_2$ articulatory representation is then described in Section 5.3, where we present some basic modelling ideas and describe the algorithms for automatically deriving the representation from the articulatory data. Experiments aimed at verifying the adequacy of the proposed representation are described in Section 5.4, which also discusses the results. We summarise the chapter in Section 5.5, describing the potential benefits of this approach and presenting some directions of future research aimed at improving the articulatory representation.

5.2 Preliminaries: The ETS$_2$ Representation Formalism

In this section we introduce the core elements of the ETS$_2$ model. In what follows, we essentially follow the white paper (Goldfarb *et al.*, 2004). However, bearing in mind the subsequent developments (Gutkin *et al.*, 2004; Gutkin and Gay, 2005b,c), which are described in the rest of this chapter, we took the decision not to go into unnecessary (for our representation) detail. The representation-specific interpretation of the concepts we introduce in this section will be given further on in Section 5.3.

5.2.1 Primitive Transformations

In this section we introduce the basic representational units of the formalism — the *primitive transformations* and *sites*, as defined in (Goldfarb *et al.*, 2004, Section 3). Informally, an ETS$_2$ primitive transformation is a unit of temporal structure (primitive event) of some natural process. This event operates on ETS$_2$ sites, which are the smallest unstructured representational units within a process. The primitive transformation can be seen to transform its set of "initial" sites into its set of "terminal" sites.

Notation 5.1. For a linearly ordered set $\mathcal{A} = \langle A, < \rangle$, the set obtained by discarding the linear order in $\mathcal{A}$ will be denoted

$$]\mathcal{A}[\stackrel{\text{def}}{=} A\,. \qquad \square$$

Definition 5.1 (Original Primitive, Site Type and Site Label)**.** Let

$$\widehat{\Pi} = \{\widehat{\pi_1}, \widehat{\pi_2}, \ldots, \widehat{\pi_n}\}$$

be a small finite set of *names of primitives*. Also let SL be a finite set of *site labels* (or simply *sites*) and ST a finite set of *site types*. Moreover, $\forall\ \widehat{\pi}_i \in \widehat{\Pi}$, we are given a triple

$$\mathring{\pi}_i = \langle \widehat{\pi}_i, \mathrm{INIT}_i, \mathrm{TERM}_i \rangle$$

called an *original primitive transformation*, or simply *original primitive*, where INIT_i and TERM_i are (small) finite, possibly empty, *linearly ordered* sets of site labels, of cardinalities k and l, respectively. For these two sets the following is true:

$$k + l \neq 0\,, \quad]\mathrm{INIT}_i[\subseteq SL \quad \text{and} \quad]\mathrm{TERM}_i[\subseteq SL\,.$$

We denote by $\mathring{\Pi}$ the finite set comprised of $\mathring{\pi}_i$, $1 \leq i \leq n$, and call it the *set of original primitives*. Finally, we are also given a *site type mapping*

$$\mathrm{TYPE} : SL \to ST\,,$$

which assigns a site type to each site label. $\square$

A site type encapsulates the inherent structural or qualitative character of a site, while site labels are merely temporary, interchangeable names. A site type specifies the kinds of allowable "interactions" of this site with the sites of other primitives. In Section 5.2.2, this point will be explained more formally and the relation between the sites and site types will become clearer.

Notation 5.2. For an original primitive $\mathring{\pi}_i$, the following concepts and notations will be useful:

$$\begin{array}{rcll} \mathrm{Init}\,(\mathring{\pi}_i) & \stackrel{\text{def}}{=} &]\mathrm{INIT}_i[& \text{is the set of } \textit{initial sites} \text{ of } \mathring{\pi}_i \\ \mathrm{Term}\,(\mathring{\pi}_i) & \stackrel{\text{def}}{=} &]\mathrm{TERM}_i[& \text{is the set of } \textit{terminal sites} \text{ of } \mathring{\pi}_i \\ \mathrm{Sites}\,(\mathring{\pi}_i) & \stackrel{\text{def}}{=} &]\mathrm{INIT}_i[\;\cup\;]\mathrm{TERM}_i[& \text{is the set of } \textit{all sites} \text{ of } \mathring{\pi}_i \\ \mathring{\pi}_i(\overline{k}) & & & \text{is the } \textit{k-th initial site} \text{ of } \mathring{\pi}_i \\ \mathring{\pi}_i(\underline{l}) & & & \text{is the } \textit{l-th terminal site} \text{ of } \mathring{\pi}_i\,. \end{array}$$

□

Pictorially, it is convenient to represent an original primitive

$$\mathring{\pi}_i \;=\; \langle \widehat{\pi}_i, \mathrm{INIT}_i, \mathrm{TERM}_i \rangle$$

as a convex shape. The initial sites are marked as points on its top, and the terminal sites are marked on its bottom. We will use numbers as labels for the sites, with the left-to-right ordering of the sites on the top and bottom corresponding to the linear orderings in INIT_i and TERM_i, respectively.

Example 5.1 (Original Primitives)**.** Four original primitives

$$\mathring{\pi}_1 \;=\; \big\langle \widehat{\pi}_1, \langle 1\rangle, \langle 1,2\rangle \big\rangle\,, \mathring{\pi}_2 \;=\; \big\langle \widehat{\pi}_2, \langle 1,2\rangle, \langle 2,3\rangle \big\rangle\,,$$
$$\mathring{\pi}_3 \;=\; \big\langle \widehat{\pi}_2, \langle 1,2\rangle, \langle 2\rangle \big\rangle \quad \text{and} \quad \mathring{\pi}_2 \;=\; \big\langle \widehat{\pi}_2, \emptyset, \langle 1,2,3\rangle \big\rangle$$

are shown in Figure 5.1. The natural numbers are used as labels incidentally, and only for convenience. Also note that the $\widehat{\ }$ symbols are dropped in this and all subsequent figures, and site types are not indicated.

In addition, alternative notation will be used in the concrete examples. Instead of $\big\langle \widehat{\pi}_i, \langle \ldots \rangle, \langle \ldots \rangle \big\rangle$, the following short notation $\widehat{\pi}_i[\,\ldots \mid \ldots\,]$ will be used (as in Example 5.2). ▷

Definition 5.2 (Site Relabelling)**.** A *site relabelling* F is an injective mapping

$$F\colon L \to SL\,, \quad \text{where} \quad L \subset SL\,,$$

which preserves site types. In other words,

$$\forall l \in L \quad \mathrm{TYPE}\,(l) = \mathrm{TYPE}\,(F(l))\,.$$ □

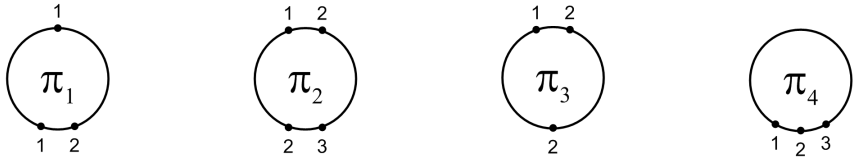


Figure 5.1: Pictorial illustration of four original primitives (reproduced with permission from Goldfarb *et al.*, 2004). Identical initial and terminal label may identify the same object on which this primitive transformation operates. Alternatively, this may be interpreted as a certain event (represented by a primitive) which occurred in a certain object's (represented by a fixed label) history.

The notion of site relabelling is crucial for introducing the concept of primitive transformations, which follows.

Definition 5.3 (Primitive)**.** For an original primitive $\mathring{\pi}_i = \langle \widehat{\pi}_i, \mathrm{INIT}_i, \mathrm{TERM}_i \rangle$, and a site relabelling of an original primitive

$$f\colon \mathrm{Sites}\,(\mathring{\pi}_i) \to SL\,,$$

primitive transformation, or simply *primitive*, is defined as

$$\mathring{\pi}_{i\{f\}} = \langle \widehat{\pi}_i, f(\mathrm{INIT}_i), f(\mathrm{TERM}_i) \rangle\,,$$

where the linear orders on $f(\mathrm{INIT}_i)$ and $f(\mathrm{TERM}_i)$ are induced by those in INIT_i and TERM_i, respectively. Correspondingly, the *set of structurally identical primitives* is defined as

$$\Pi_i \stackrel{\mathrm{def}}{=} \left\{ \mathring{\pi}_{i\{f\}} \mid f \text{ is an original primitive site relabelling } \right\}$$

and the set of all primitives is given by

$$\Pi \stackrel{\mathrm{def}}{=} \bigcup_{i=1}^{n} \Pi_i\,.$$

Primitives $\pi_i, \pi_j \in \Pi$ are *structurally identical* if there exists a primitive site relabelling

$$f\colon \mathrm{Sites}\,(\pi_i) \to SL \quad \text{such that} \quad \pi_j = \pi_{i\{f\}}\,.$$

The corresponding equivalence class Π_j will be called *class primitive* and denoted $[\pi_j]$. □

Example 5.2 (Primitives and Class Primitives)**.** Figure 5.2 shows primitives

$$\widehat{\pi}_1[\,5\,|\,5,3\,]\,,\ \widehat{\pi}_2[\,3,4\,|\,4,6\,]\,,\ \widehat{\pi}_2[\,1,2\,|\,2,3\,] \quad \text{and} \quad \widehat{\pi}_2[\,4,3\,|\,3,5\,]\,.$$

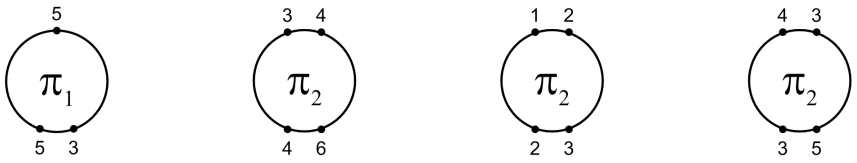


Figure 5.2: Pictorial illustration of four primitives (reproduced with permission from Goldfarb *et al.*, 2004).

Figure 5.3: Pictorial illustration of two class primitives (reproduced with permission from Goldfarb *et al.*, 2004).

Note that the last three primitives are instances of the same class primitive $[\pi_2]$. The notion of site labels allows one to differentiate between various instances of the same class primitive. By introducing the notion of site relabelling these instances can be related to the same class of primitives if they are structurally identical under the given relabelling (Definition 5.2).

The two class primitives (unrelated to the primitives in Figure 5.2) are shown in Figure 5.3. The circle and the square denote two distinct site types. This implies that letters $\{a, b\}$ and $\{x, y\}$ are the names of the variables that are allowed to vary over *non-overlapping* sets of numeric labels. ▷

Remark 5.1 (Note on relabellings)**.** From this point onwards, we omit the notion of the relabelling from the discussion for the sake of brevity. In general, all the structural concepts introduced in the next sections formally allow for relabellings introduced on their constituent sets of sites. Since in our representation (which will be discussed further on in Section 5.3) the interactions between the sites are *formally* quite simple and can be expressed by the identity mapping, we decided not to over-burden the exposition here.

5.2.2 Instances of Structural History

An ETS$_2$ *struct* is a temporally-ordered sequence of connected primitives capturing an instance of a "structural history" of the corresponding process or object. The structs were defined in (Goldfarb *et al.*, 2004, Section 4). The definition of a struct can be

seen as a *structural* generalisation of the inductive process of construction of natural numbers, proposed by Giuseppe Peano (Landau, 1951).

Definition 5.4 (Struct)**.** The set Σ of *instances of structural history*, or simply *structs*, is defined inductively follows: For each $\sigma \in \Sigma$, three sets — $\mathrm{Init}\,(\sigma)$, $\mathrm{Term}\,(\sigma)$, and $\mathrm{Sites}\,(\sigma)$ of *initial sites*, *terminal sites*, and *all sites* of the struct σ — are inductively constructed:

- θ is the *null struct* whose sets of sites are

$$\mathrm{Init}\,(\theta) \;=\; \mathrm{Term}\,(\theta) \;=\; \mathrm{Sites}\,(\theta) \;\stackrel{\text{def}}{=}\; \varnothing$$

- Assuming that $\sigma \in \Sigma$ has been constructed, and given $\pi \in \Pi$ satisfying

$$\mathrm{Sites}\,(\sigma) \cap \mathrm{Sites}\,(\pi) \;=\; \mathrm{Term}\,(\sigma) \cap \mathrm{Init}\,(\pi)\,, \tag{5.1}$$

  the expression

$$\sigma \dashv \pi$$

  signifies the new struct σ_π, called the *continuation of struct σ by primitive π*. The sets of sites of σ_π are constructed as follows:

$$\mathrm{Init}\,(\sigma_\pi) \;\stackrel{\text{def}}{=}\; \mathrm{Init}\,(\sigma) \cup [\mathrm{Init}\,(\pi) \setminus \mathrm{Term}\,(\sigma)] \tag{5.2}$$

$$\mathrm{Term}\,(\sigma_\pi) \;\stackrel{\text{def}}{=}\; \mathrm{Term}\,(\pi) \cup [\mathrm{Term}\,(\sigma) \setminus \mathrm{Init}\,(\pi)] \tag{5.3}$$

$$\mathrm{Sites}\,(\sigma_\pi) \;\stackrel{\text{def}}{=}\; \mathrm{Sites}\,(\sigma) \cup \mathrm{Sites}\,(\pi)\,. \tag{5.4}$$

The operation $\dashv$ is called the *continue operation*. The struct σ is specified by the following expression encapsulating its construction process

$$\sigma \;=\; [\pi_1 \dashv \pi_2 \dashv \cdots \dashv \pi_t]\,.$$

The order relationship between the indices in the above expression corresponds to the constructive order of the relevant continue operations. It is assumed that this expression is valid for $t = 0$ and, in this case, denotes θ. □

For the above construction of $\sigma \dashv \pi$, the continuation operation $\dashv$ can be depicted and thought of as an *attachment* of the identical sites in $\mathrm{Term}\,(\sigma)$ and $\mathrm{Init}\,(\pi)$. In particular, primitive π *is attached to* primitive π_i if, when actually constructing $\sigma \dashv \pi$, *at least one* initial site of π was attached to one terminal site of π_i.

Example 5.3 (Struct)**.** Two structs, where the set of original primitives includes $\mathring{\pi}_1, \mathring{\pi}_2, \mathring{\pi}_3, \mathring{\pi}_4, \mathring{\pi}_5$ are shown in Figure 5.4. The vertical order of primitives corresponds to the constructive (temporal) order of the relevant continue operations. ▷

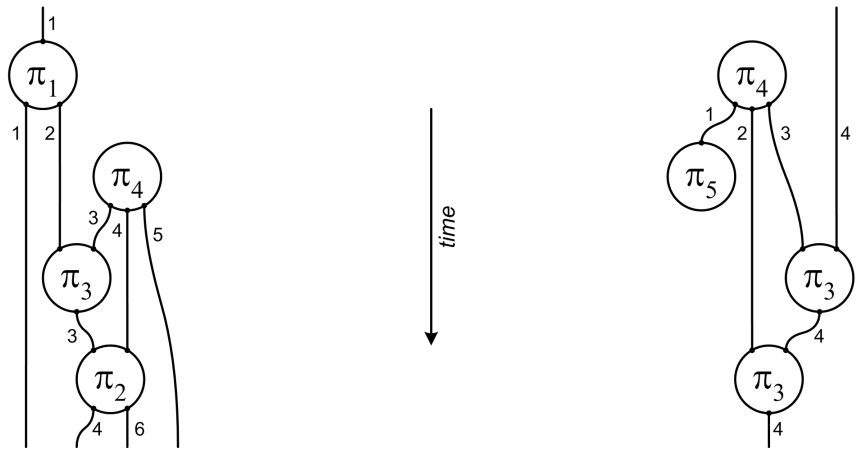


Figure 5.4: Two instances of structural history (structs) (reproduced with permission from Goldfarb *et al.*, 2004). Structs can be informally interpreted as evolving sequences of events (primitive transformations) sharing some attributes (sites). The application-specific interpretation of structural histories is provided later on in this chapter in Section 5.3.

The following definition introduces the formal procedure for composing several instances of structural histories:

Definition 5.5 (Struct Composition)**.** Let α and β be structs such that

$$\mathrm{Init}\,(\beta) \subseteq \mathrm{Term}\,(\alpha)\,.$$

If the following inductive construction procedure, denoted by $\lhd$, can be completed

$$\begin{array}{lll} \text{for} \quad \beta = \theta\colon & \alpha \lhd \theta & \stackrel{\text{def}}{=} \quad \alpha\,; \\ \text{for} \quad \beta = \gamma \dashv \pi\colon & \alpha \lhd (\gamma \dashv \pi) & \stackrel{\text{def}}{=} \quad (\alpha \lhd \gamma) \dashv \pi \end{array}$$

then the resulting struct

$$\alpha \lhd \beta$$

is called the *composition* of α and β and we say that β *is composable with* α. □

Lemma 5.1. *The sets of sites for the composition of two structs* α *and* β *are given by*

$$\begin{array}{rcl} \mathrm{Init}\,(\alpha \lhd \beta) & = & \mathrm{Init}\,(\alpha) \cup \big[\mathrm{Init}\,(\beta) \setminus \mathrm{Term}\,(\alpha)\big] \\ \mathrm{Term}\,(\alpha \lhd \beta) & = & \big[\mathrm{Term}\,(\alpha) \setminus \mathrm{Init}\,(\beta)\big] \cup \mathrm{Term}\,(\beta) \\ \mathrm{Sites}\,(\alpha \lhd \beta) & = & \mathrm{Sites}\,(\alpha) \cup \mathrm{Sites}\,(\beta)\,. \end{array}$$

Note that not every two structs satisfying $\mathrm{Init}\,(\beta) \subseteq \mathrm{Term}\,(\alpha)$ are composable, as demonstrated by the following example.

Example 5.4 (Struct Composition)**.** Two structs (α and β) and their composition ($\alpha \lhd \beta$) are shown in Figure 5.5. Note that $\beta \lhd \alpha$ is not a legal composition. ▷

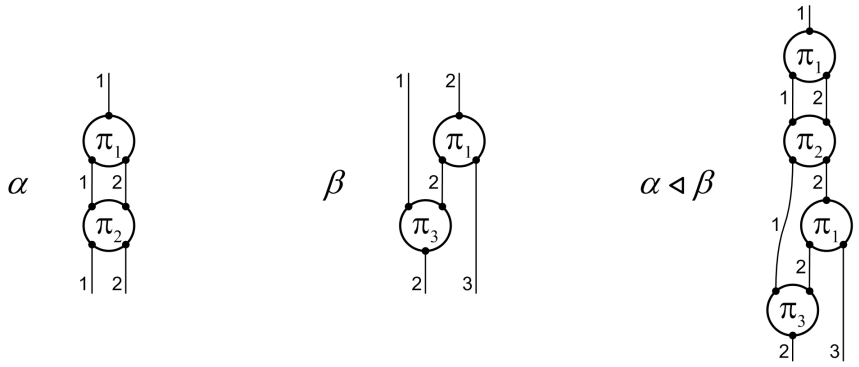


Figure 5.5: Two structs (α and β) and their composition $\alpha \triangleleft \beta$ (reproduced with permission from Goldfarb *et al.*, 2004).

5.2.3 Extructs

Before describing the central concepts of the model in the next section, in this section we introduce an important auxiliary notion of an *extruct*, as defined in (Goldfarb *et al.*, 2004, Section 5). Given a certain instance of structural history of some process under investigation, it is often desirable to be able to examine some *recent* fragment of this structure. This is especially useful if one is expecting a certain (non-trivial) event to appear in the structure. The expected appearance of this event is signalled by the presence in the struct of some structure (context) which is described by an extruct (this will become clearer in Section 5.2.4).

Definition 5.6 (Attachment Graph)**.** The *attachment graph* for struct

$$\sigma = [\pi_1 \dashv \pi_2 \dashv \cdots \dashv \pi_t]$$

is defined as the following directed graph:

$$G_\sigma = \langle V_\sigma, E_\sigma \rangle ,$$

where

$$V_\sigma = \{v_1, v_2, \cdots, v_t\}, \quad v_i \text{ corresponds to } \pi_i$$

and $\langle v_i, v_j \rangle \in E_\sigma$ if in the inductive construction of σ, π_j was attached to π_i. □

The concept of attachment graph is a simplified partial encapsulation of the notion of a struct. Multiple attachments between any pair of primitives are recorded as a single edge, as demonstrated in Figure 5.6. Next, we define a concept of an *interfaced struct*. Informally, an interfaced struct is pair consisting of a struct and some subset of its terminal sites.

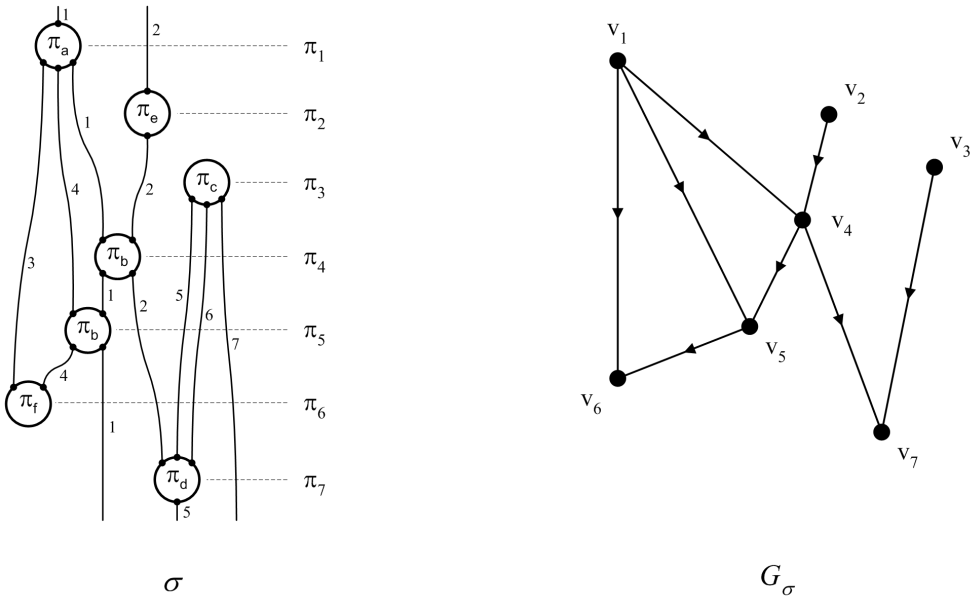


Figure 5.6: A simple struct (left) and the corresponding attachment graph (right) (reproduced with permission from Goldfarb *et al.*, 2004).

Definition 5.7 (Interfaced Struct)**.** An *interfaced struct* is a pair $\langle \sigma, \text{Iface} \rangle$, where σ is a struct

$$\sigma \quad = \quad [\pi_1 \dashv \pi_2 \dashv \cdots \dashv \pi_t]$$

and Iface is a subset of $\text{Term}(\sigma)$ called the *set of interface sites*. For each primitive π_i in the above σ, $1 \leq i \leq t$, a *constituent of* $\langle \sigma, \text{Iface} \rangle$ is the following 4-tuple

$$\mathfrak{e}^i_\sigma \quad \stackrel{\text{def}}{=} \quad \langle \pi_i, \text{DIS}_i, \text{DTS}_i, \text{IS}_i \rangle \,,$$

where

- DIS_i is the set of *detached initial sites*, $\text{DIS}_i \subseteq \text{Init}(\pi_i)$, consisting of those initial sites that are not attached to any other primitive;
- DTS_i is the set of *detached terminal sites*, $\text{DTS}_i \subseteq \text{Term}(\pi_i) \setminus \text{Iface}$, consisting of those terminal sites that are not attached to any other primitive;
- IS_i is the set of *interfaced sites*, $\text{IS}_i \subseteq \text{Term}(\pi_i) \cap \text{Iface}$, consisting of those terminal sites that are not attached to any other primitive. □

Definition 5.8 (Extruct (Informal Definition))**.** Informally, an *extruct* is a 3-tuple

$$\varepsilon_\sigma \stackrel{\text{def}}{=} \langle \sigma, \text{Iface}, \mathfrak{E} \rangle \,,$$

where σ and Iface form an interface struct (Definition 5.7) and the set $\mathfrak{E}$ consists of those constituents $\mathfrak{e}^i_\sigma$ of $\langle \sigma, \text{Iface} \rangle$ whose sites attach (directly or indirectly) to primitives with the interface sites. □

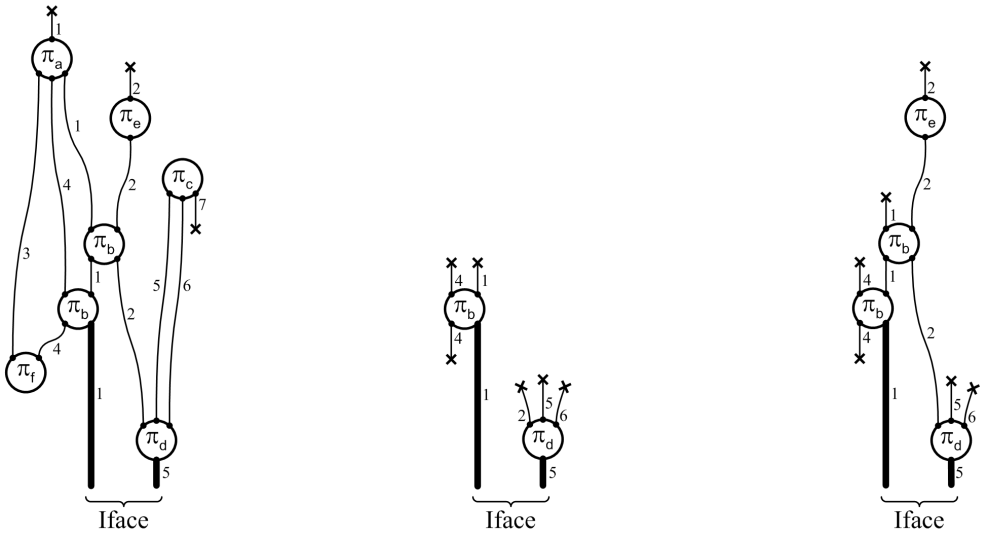


Figure 5.7: Some of the possible extructs corresponding to the struct shown in Figure 5.6 (reproduced with permission from Goldfarb *et al.*, 2004).

For more details on construction procedure for the set $\mathfrak{E}$ above, refer to (Goldfarb *et al.*, 2004, Section 5). In general, each interface struct has multiple corresponding extructs, the incremental construction of which involves bottom-up traversal of the struct and the corresponding attachment graph.

Example 5.5 (Extruct)**.** Figure 5.7 shows some of the possible extructs for an interfaced struct $\langle \sigma, \text{Iface} \rangle$ corresponding to an actual struct σ shown in Figure 5.6. Heavy lines identify interface sites, crosses identify detached initial and terminal sites. ▷

5.2.4 Transformations and Supertransformations

In this section we introduce the central structural units of the model: the transform and the supertransform (Goldfarb *et al.*, 2004, Section 6). Informally, it is useful to think of every struct (Section 5.2.2) as being formed (generated) by a series of non-trivial structural units — transformations.

Definition 5.9 (Transform)**.** A *transformation*, or simply *transform*, is a pair

$$\tau = \langle \varepsilon, \beta \rangle ,$$

where extruct $\varepsilon = \langle \text{Iface} , \mathfrak{E} \rangle$ and struct β satisfy

$$\text{Iface} = \text{Init} (\beta) = \text{Sites} (\varepsilon) \cap \text{Sites} (\beta) .$$

We call ε the *context of transform* τ, denoted $\text{cntx}(\tau)$, and β the *body of transform* τ, denoted $\text{body}(\tau)$. The *set of all sites of transform* τ is defined as

$$\text{Sites} (\tau) \stackrel{\text{def}}{=} \text{Sites} (\varepsilon) \cup \text{Sites} (\beta) . \qquad \square$$

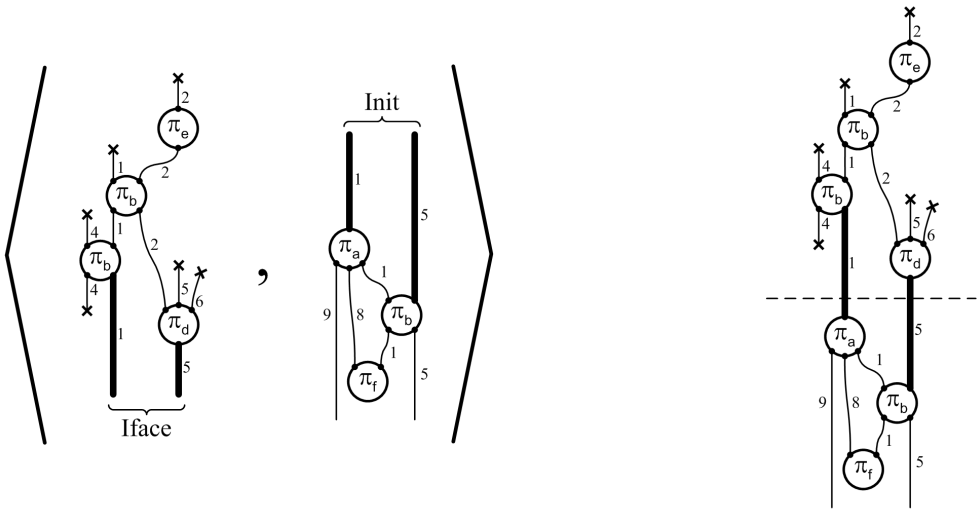


Figure 5.8: An example of a transform whose context corresponds to one of the extructs shown in Figure 5.7. The right hand side depicts the "assembled" transform corresponding to a more appropriate interpretation/understanding of the transform (reproduced with permission from Goldfarb *et al.*, 2004).

Example 5.6 (Transform)**.** Figure 5.8 shows a simple transformation whose context corresponds to one of the extructs shown in Figure 5.7. ▷

Since every struct can be seen as representing some non-trivial object or event in the application domain, it clearly belongs to some *class* of objects or events. Hence, in theory one can describe the class structure by enumerating all the structs (samples of that class) in the domain. There is a better option, however. Rather than using structs in the class description, it is more economical to use the transforms which generate these structs. This observation leads to the notion of *supertransform*, which can be seen as a generalisation of a transformation concept.

Definition 5.10 (Supertransform)**.** A *supertransformation*, or simply *supertransform*, is a pair

$$\boldsymbol{\tau} \stackrel{\text{def}}{=} \langle E, B \rangle \,,$$

where

$$E = \left\{\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_p\right\} \quad \varepsilon_i = \langle \text{Iface}\,_i, \mathfrak{E}_i \rangle, \quad B = \left\{\beta_1, \beta_2, \cdots, \beta_q\right\},$$

if the following conditions hold

$$\forall\, i, j, k \qquad \begin{aligned} \text{Init}\,(\beta_i) = \text{Init}\,(\beta_j) = \text{Iface}\,_k = \text{Sites}\,(\varepsilon_k) \cap \text{Sites}\,(\beta_i) \\ \text{Term}\,(\beta_i) = \text{Term}\,(\beta_j)\,. \end{aligned}$$

The *constituent transform set* for a supertransform $\boldsymbol{\tau}$ is defined as the set of all transforms specified by the elements of the Cartesian product $E \times B$. It is convenient

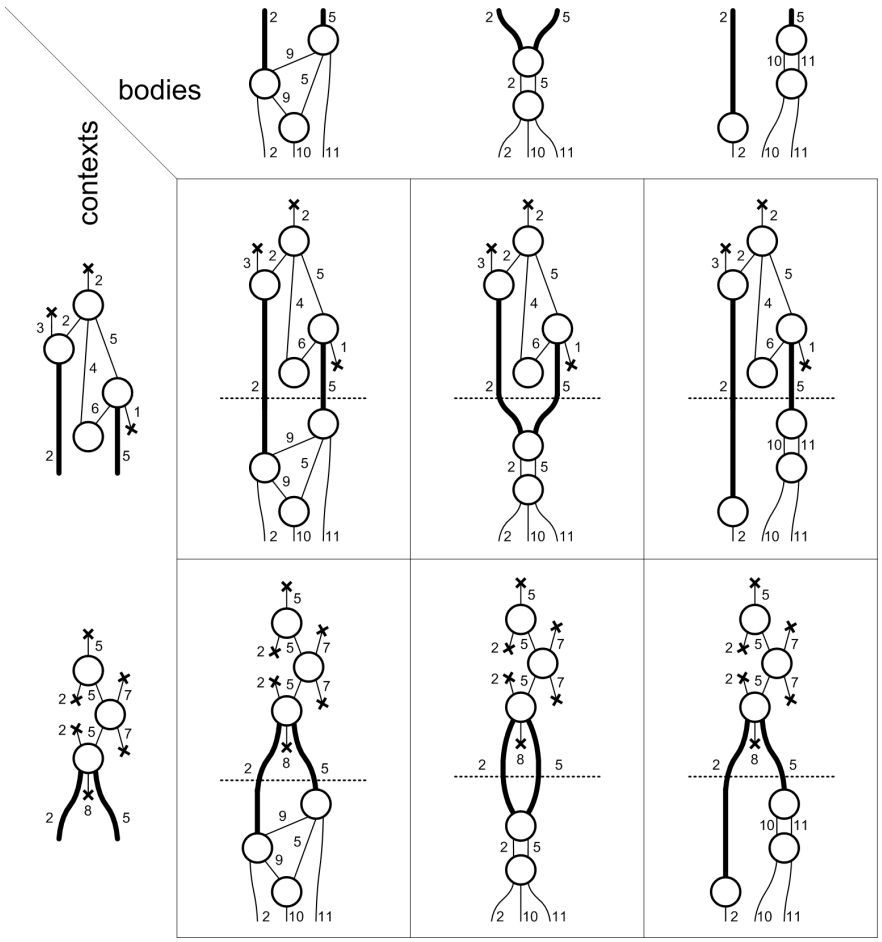


Figure 5.9: Visualisation of a supertransform. Note that all contexts have the same interface sites and all bodies have the same initial and terminal sites (reproduced with permission from Goldfarb *et al.*, 2004).

to blur the distinction between the pair $\langle E, B\rangle$ and the product $E \times B$, and to refer to both of them as the supertransform $\boldsymbol{\tau}$. Thus the following notation will be used: $\tau = \langle \varepsilon, \beta\rangle$, $\tau \in \boldsymbol{\tau}$. □

Example 5.7 (Supertransform)**.** A simple supertransform consisting of six constituent transformations is shown in Figure 5.9 as a rectilinear table. All contexts have the same interface sites and all bodies have the same initial and terminal sites. ▷

If there are multiple site relabellings used in the representation, one can generalise the notion of a supertransform $\boldsymbol{\tau}$ to that of a *class supertransform*, denoted $[\boldsymbol{\tau}]$, which is defined as an equivalence class on the set of all supertransforms which are equivalent under the relabelling. In the representation dealt with in this chapter, we can treat the notions of class supertransform and supertransform as equivalent (see Remark 5.1).

The concept of a supertransform is central in ETS$_2$ formalism, because it encapsulates the *structural* means of class description. Given a supertransform, one can generate

(or recognise) an infinite number of objects (structs) of that class. This is because the structs are generated by the constituent transforms of a supertransform.

5.2.5 Level Ascension Postulate

Perhaps the most powerful feature of the ETS$_2$ formalism is its ability to model the environment at multiple levels. Within the ETS$_2$ model, the transition to a new level of representation consists of construction of a new next-level set of primitives, which can then be used constructively in the usual manner (as described in the previous sections) to construct the set of next-level structs, extructs, transforms and so on. The corresponding formal machinery was defined in (Goldfarb *et al.*, 2004, Section 8).

Proposition 5.1 (Level Ascension Postulate)**.** *The class of (context-sensitive) macro-events corresponding to a class supertransform may be adequately represented at the next level by a new (original) primitive obtained by completely shrinking that supertransform's contexts and by dropping the internal structure of the supertransform's bodies in the manner described in Definition 5.11.* □

The following definition is a direct consequence of the above postulate and Definition 5.1 (including the notation in the definition).

Definition 5.11 (Next-Level Correspondence)**.** Assume that we have fixed a set TS of class supertransforms,

$$\mathrm{TS} = \left\{ [\boldsymbol{\tau}_1], [\boldsymbol{\tau}_2], \cdots, [\boldsymbol{\tau}_m] \right\},$$

called a *transformation system*. Define three sets

$$\begin{aligned}
\widehat{\Pi}' &\stackrel{\text{def}}{=} \left\{ \widehat{[\boldsymbol{\tau}_1]}, \widehat{[\boldsymbol{\tau}_2]}, \cdots, \widehat{[\boldsymbol{\tau}_m]} \right\} && \text{of } \textit{next-level primitive names}\,, \\
SL' &\stackrel{\text{def}}{=} SL && \text{of } \textit{next-level site labels}\,, \\
ST' &\stackrel{\text{def}}{=} ST && \text{of } \textit{next-level site types}\,.
\end{aligned}$$

We now introduce a set of *next-level original primitives* $\mathring{\Pi}'$ for which each of its elements $\mathring{\pi}'_i$ is constructed as follows:

$$\mathring{\pi}'_i \stackrel{\text{def}}{=} \left\langle \widehat{[\boldsymbol{\tau}_i]}, \mathrm{INIT}_i, \mathrm{TERM}_i \right\rangle$$

where, for

$$\boldsymbol{\tau}_i = \langle E_i, B_i \rangle \quad \text{with} \quad B_i = \left\{ \beta_{i_1}, \beta_{i_2}, \ldots, \beta_{i_{q_i}} \right\},$$

$$\begin{aligned}
]\mathrm{INIT}_i[&\stackrel{\text{def}}{=} \mathrm{Init}\left(\beta_{i_1} \right), \\
]\mathrm{TERM}_i[&\stackrel{\text{def}}{=} \mathrm{Term}\left(\beta_{i_1} \right)
\end{aligned}$$

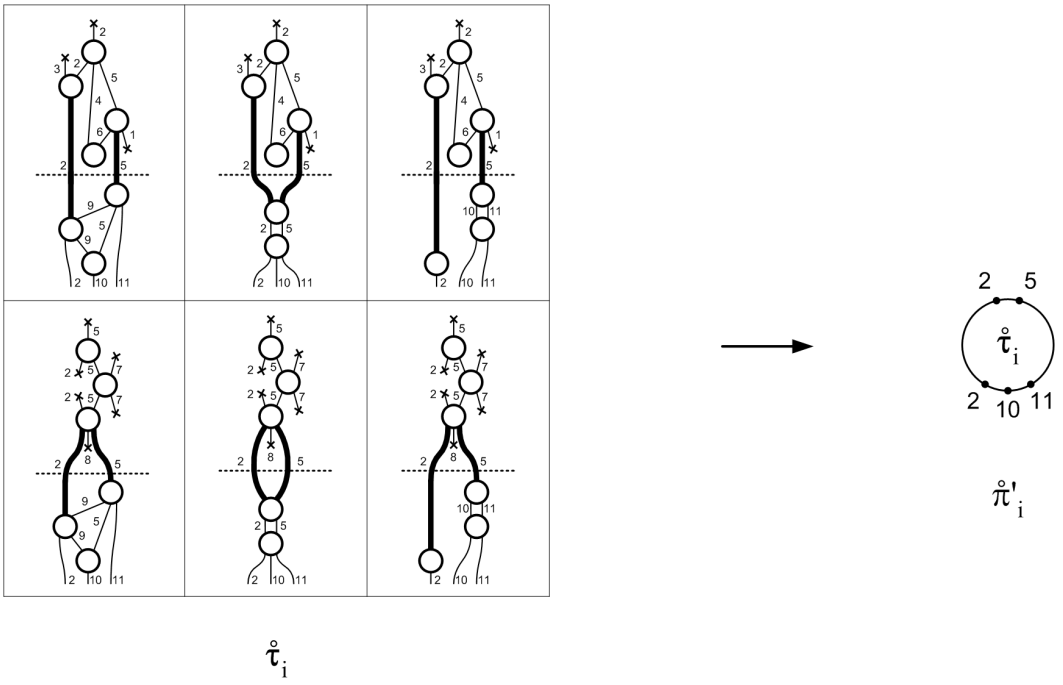


Figure 5.10: Some supertransform $\mathring{\tau}'$ and the corresponding next-level original primitive (reproduced with permission from Goldfarb *et al.*, 2004). The symbol $\widehat{\ }$ in the depiction of the next-level original primitive is dropped.

and the corresponding linear orders are induced based on both the constructive order of the primitives in the first body of $\boldsymbol{\tau}_i$ as well as on the orders of the sites in each of those primitives (see Figure 5.10). In addition, we define a *next-level site type mapping*

$$\text{TYPE}' \colon SL' \to ST'$$

to be the same as mapping TYPE in Definition 5.1. □

The next-level original primitive name

$$\widehat{[\boldsymbol{\tau}_i]}\,,\ 1 \le i \le m$$

in the above definition could be thought of as denoting the "name" given to a class supertransform $[\boldsymbol{\tau}_i]$, which is inherited by the next-level original primitive. The next-level counterparts of the notions described in the previous sections (primitives, structs, transforms and supertransforms) are described in exactly the same manner as before using the next-level original primitives.

5.2.6 Inductive Structure

Finally, we can encapsulate the entire developed mathematical structure as a single entity in the following definition (Goldfarb *et al.*, 2004, Section 8):

Definition 5.12 (Inductive Structure)**.** A *(single-level) inductive structure* is a pair

$$\langle \mathring{\Pi}, \mathrm{TS} \rangle ,$$

where $\mathring{\Pi}$ is a set of original primitives and TS is a transformation system. The latter pair also signifies all relevant concepts, such as structs, extructs, and so on.

A *multi-level inductive structure* (with l levels) MIS is an l-tuple

$$\mathrm{MIS} \stackrel{\mathrm{def}}{=} \left\langle \langle \mathring{\Pi}, \mathrm{TS} \rangle, \langle \mathring{\Pi}', \mathrm{TS}' \rangle, \cdots, \langle \mathring{\Pi}^{(l-1)}, \mathrm{TS}^{(l-1)} \rangle \right\rangle$$

where $\mathrm{TS}^{(l-1)} = \varnothing$, $\mathrm{TS}^{(k)}$ is the transformation system for the set of original primitives $\mathring{\Pi}^{(k)}$, and every consecutive pair of inductive structures satisfies the level ascension postulate from the previous section (see Figure 5.11 and Figure 5.12).

Every k-th level inductive structure $\left\langle \mathring{\Pi}^{(k)}, \mathrm{TS}^{(k)} \right\rangle$ in MIS is denoted

$$\mathrm{MIS}(k) \stackrel{\mathrm{def}}{=} \langle \mathring{\Pi}^{(k)}, \mathrm{TS}^{(k)} \rangle \qquad k = 0, 1, \ldots, l-1 .$$

In addition,

$$\boldsymbol{\tau}^{(k)} \rightarrow \pi^{(k+1)} \qquad k = 0, 1, \ldots, l-2$$

denotes the transition from some supertransform $\boldsymbol{\tau}^{(k)}$ at level k to a corresponding primitive $\pi^{(k+1)}$ at level $k+1$. □

Learning in ETS$_2$ formalism reduces to the discovery of the inductive class structure outlined above. The learning algorithm is presented in Part III of the white paper (Goldfarb *et al.*, 2004) and is not, strictly speaking, part of the formalism. Because the algorithm is rather involved, for the sake of brevity we decided not to present it in this thesis, although it has been successfully implemented and tested on the articulatory structures (Section 5.3) derived from the real data. Briefly, the algorithm attempts to optimally capture the class representation of the environment by expanding and refining its multi-level inductive structure MIS, including the number of its levels. This is accomplished by the creation and modification (but never deletion) of relevant class supertransforms at the appropriate levels. This is mainly achieved by the introduction of the numeric components associated with the main structural concepts presented above. Weights are added to the following structural associations:

- Connections between primitives;
- Connections between bodies and contexts of the transforms;
- Connections between primitives and structs;
- Constituent transforms of the supertransforms.

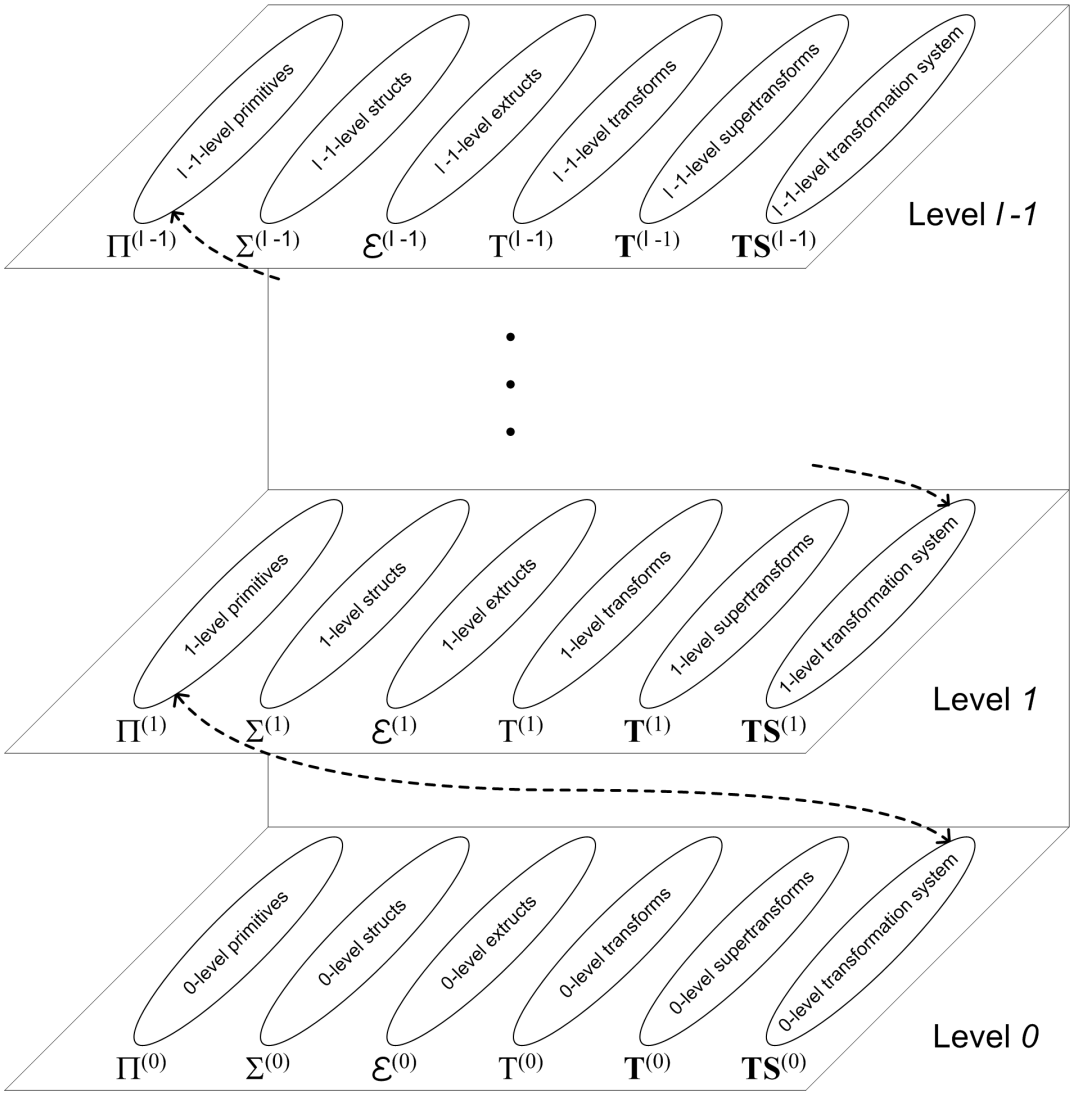


Figure 5.11: Schematic representation of a multi-level inductive structure with l levels (reproduced with permission from Goldfarb *et al.*, 2004).

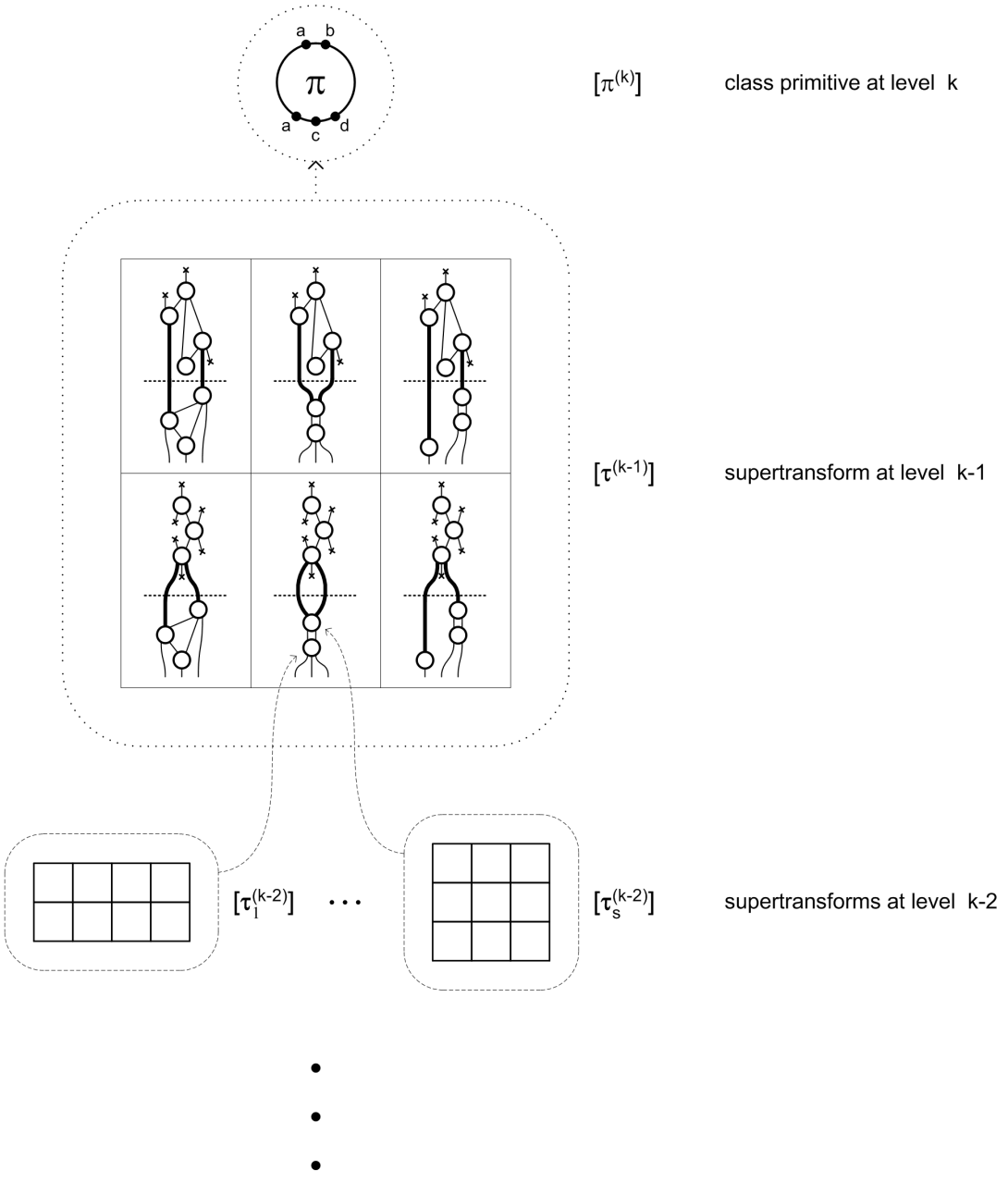


Figure 5.12: Pyramid view (partial) of a k-th level class supertransform: the pyramid should be thought of as being formed by the subordinate class supertransforms (reproduced with permission from Goldfarb *et al.*, 2004).

These weights are related to the statistical observations of the above associations in the structs derived by the pre-processor from the data. Numeric association schemes allow to establish that some structures are more likely to be observed than others. Once the weights are updated, the hybrid numeric-structural algorithm updates (or creates) the relevant structures (extructs, transforms and supertransforms), spawning new levels if necessary. In general, the learning proceeds on all the current levels of the hierarchy.

5.3 Articulatory Representation

In Section 5.3.1, we explain the basic tenets of the articulatory representation and set the scene for subsequent developments. The section also presents an answer to a question of how to formally approach the modelling of articulatory structures within ETS_2. We also present some representation-specific assumptions which make the modelling conceptually simpler.

Before proceeding with the representation, in Section 5.3.2 we introduce the articulatory speech corpus used in this study. Our main reason for introducing it here, rather than in the experimental section, is simple. Our goal is: on the one hand to present a formal articulatory representation, and on the other, to show how this formal model is related to real measurements. Moreover, in later sections of this chapter we show how to automatically derive this model from the data.

Having introduced the corpus, in Section 5.3.3 we present the atomic units of the representation — the primitive gestures, which are formally treated as ETS_2 primitives. Next, in Section 5.3.4 we present a conceptually simple procedure (first suggested in Gutkin and King, 2005a) for derivation of the primitive gestures directly from the articulatory corpus at hand.

We next address the issue of class representation of consonantal phonemes of English within the ETS_2 formalism. In Section 5.2 we mentioned that this can be achieved by designing an articulatory representation via the ETS_2 transforms and supertransforms. In particular, the transform allows us to describe a particular pattern of constriction and release of the consonantal sound, while the supertransform encapsulates the family of these semantically and structurally related patterns (transforms). These non-trivial structural units of representation are described in Section 5.3.6 and Section 5.3.7. The latter sections are based on the early ideas on the representation which we reported in (Gutkin *et al.*, 2004) and extended in (Gutkin and Gay, 2005c).

In Section 5.3.8, we describe a structural search procedure for locating the constituent transforms of a given supertransform in any given struct generated from the real data. The section is based on our recent work, reported in (Gutkin and Gay,

2005a,b).

In this chapter, we focus on an initial (also called *sensory*) level of representation. This level is called sensory because it directly interacts with the data. The mechanism for extending the representation to higher levels is briefly outlined in Section 5.3.9.

5.3.1 Primitive Articulatory Gesture

In line with the process, event-based, philosophy of the ETS_2 formalism presented in Section 5.2, we base our analysis on the various articulatory processes (gestural events and combinations thereof), which operate and cause changes in the states of the articulatory organs. Various dynamic interactions[1] between the articulatory organs during the articulation process are represented as ETS_2 primitive transformations, which we introduced in Section 5.2.1.

The choice of initial level ETS_2 primitives therefore amounts to the human expert performing the following tasks:

1. Identifying the articulators participating in speech production based on physiological (Kaplan, 1971; Zemlin, 1968) and phonetic (Ladefoged, 2001) evidence. These articulators are chosen to correspond to the *site types* of the primitives.

2. Selecting the most distinct gestures involving the articulators specified above. These gestures are chosen to be the *names* of the ETS_2 primitives. Hence, gestures can be seen as transformations of the articulators.

The theoretical motivation for this more general outlook on the articulators and the interactions between them is supported by linguistic theory, which states that an articulatory analysis on a physiologically lower, motor level introduces too much anatomical detail that is linguistically irrelevant for the discrimination between various sound patterns (Ladefoged, 2001).

The following important domain-specific observations lead to a certain simplification of the formal structure of primitive transformations within our representation:

1. Although the articulators share some mechanical degrees of freedom, they are commonly assumed to be anatomically distinct and independent. In other words, any constriction formed by one of the organs does not necessarily produce a constriction in any other (Goldstein and Fowler, 2003). In our representation, this is reflected in the choice of the sites of the primitives. Any primitive in the representation possesses one specific property: it does not have multiple sites of the same type.

[1] Which phoneticians also call *processes*, e.g. a nasal sound is a result of an "oro-nasal process".

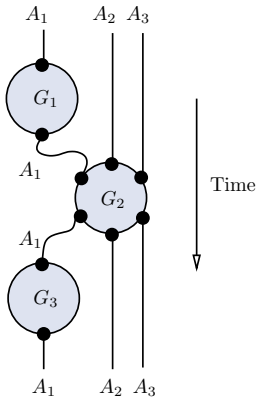


Figure 5.13: Pictorial view of an abstract gestural structure.

2. We also observe that the number and type of the articulatory organs involved in the production of any given gesture do not change with time. This leads to an important simplifying assumption that the sets specifying the initial and terminal sites of each primitive are identical.

The above linguistically-valid assumptions simplify various technical issues involved in the ETS$_2$ representation. In particular, for the site type mapping TYPE (Definition 5.1), which, given a site label, assigns to it a corresponding type, one can now use a simple one-to-one mapping. Without a loss of generality, this allows us to use site types instead of site labels in all the figures which follow.

Example 5.8 (Abstract Articulation)**.** Figure 5.13 shows an abstract articulation involving articulatory organs A_1, A_2, A_3 and three gestures G_1, G_2 and G_3 making use of these organs. The vertical positioning of the gestures corresponds to the actual flow in time of the pre-processing algorithm which detects them. The gesture G_1 operates on one articulator A_1 only, whereas gesture G_2 involves all of the depicted articulators and follows G_1. Gesture G_1 might mean "raise A_1", gesture G_2 might mean "move A_1 to A_2 while A_3 vibrates", while gesture G_3 could mean "lower A_1".

Within the ETS$_2$ formalism, this pictorial representation corresponds to the temporal sequence of three *primitives* $G_1[A_1|A_1]$, $G_2[A_1, A_2, A_3|A_1, A_2, A_3]$ and $G_3[A_1|A_1]$ which form a *struct* $G_\sigma = [G_1 \dashv G_2 \dashv G_3]$ representing some non-trivial gesture. The structs were introduced in Section 5.2.2. ▷

It is not difficult to see that each gesture, represented by an ETS$_2$ primitive, encapsulates both syntactic and semantic information. The syntactic information, allows for structural processing by the appropriate training and recognition algorithms defined within the ETS$_2$ framework (Goldfarb, 2004; Goldfarb *et al.*, 2004), while the semantic information makes the representation meaningful and fully interpretable.

5.3.2 The Articulatory Corpus

Speech corpora containing articulatory measurements are becoming quite popular with the automatic speech recognition community as more researchers become interested in using articulatory parameters either as a supplement to or substitute for spectrally based input parameters, or as an internal representation for the model[2]. The discussion of various articulatory approaches to statistical ASR is outside the scope of this thesis and we refer the interested reader to an overview by Richmond (2001).

The articulatory corpus we are using is the MOCHA corpus (Wrench, 2000; Wrench and Hardcastle, 2000). The MOCHA corpus consists of articulatory and acoustic recordings of 460 phonetically-rich sentences designed to provide good phonetic coverage of English. At the moment, the database contains the finalised recordings for one male and one female speaker, each consisting of approximately 31 minutes of speech. The particular datasets we used came from the recording of a female (acronym `fsew`) and male (acronym `msak`) speaker of British English.

The articulatory channels include Electromagnetic Articulograph (EMA) sensors directly attached to the upper and lower lips, lower incisor (jaw), tongue tip (5-10 mm from the tip), tongue blade (approximately 2-3 cm posterior to the tongue tip sensor), tongue back (dorsum) (approximately 2-3 cm posterior to the tongue blade sensor) and soft palate (velum). The EMA data has been recorded at 500 Hz. Coils attached to the bridge of the nose and the upper incisor provided the frame of reference.

Laryngograph/EGG measures changes in the contact area of the vocal folds, providing the recording of the laryngeal waveform. Pitch and voicing information can be derived from the laryngeal waveform exactly in the same fashion as from the acoustic waveform, which is also provided by the corpus. Both the laryngeal and acoustic waveforms were recorded at 16 kHz.

Electropalatograph (EPG) measurements provide tongue-palate contact data at 62 normalised positions across the hard palate (Wrench, 2000). EPG information is very useful because it augments some of the information missing from the EMA data. The EPG measurements are produced by the subject wearing an artificial palate specially moulded to fit their hard palate with the 62 electrodes mounted on the surface to detect lingual contact. Each EPG frame (the EPG.3 version of the device was used), sampled at 200 Hz, consists of 64 bits, two bits of which are unused. Each bit from the 62 bit mask is on if the contact was detected, off otherwise.

The articulatory data was post-processed to synchronise the channels and correct for the EMA head movement and discrepancies in coil placements during the recording.

[2]Traditionally, articulatory research received more attention from the linguistic community (Byrd, 2003; Perkell, 1969).

Organ	Semantics	Measurement Type
UL	upper lip	EMA
LL	lower lip	EMA
UI	upper incisor	EMA
TD	tongue back (dorsum)	EMA, EPG
TT	tongue tip	EMA, EPG
VL	soft palate (velum)	EMA, EPG
HP	hard palate	EPG
AR	alveolar ridge	EPG
VF	vocal folds	laryngeal, acoustic

Table 5.1: Articulators involved in the production of primitive gestures and the types of available measurements.

The resulting coordinate system of EMA trajectories consisting of (x, y) coordinates has its origin at the bridge of the nose, with positive x direction being towards the back of the vocal tract, away from the teeth, and positive y direction being upwards towards the roof of the mouth. The post-processing step details can be found in (Richmond, 2001).

The corpus was automatically labelled using forced alignment of the acoustic signal with phone sequences generated from a phonemic dictionary, thus phonetic labels are available (see Wrench, 2000; Wrench and Hardcastle, 2000 for more information). The autolabelling errors were hand-corrected.

5.3.3 Primitive Gestures and Their Groups

Table 5.1 lists all of the ETS$_2$ site types (articulatory organs) used in our representation. Along with each site type we show the corresponding interpretation and the source of measurements offered by MOCHA. As can be see from Table 5.1, for some articulators (like tongue tip), several sources of measurement are available.

In Section 5.3.1 we mentioned that due to the assumption that the articulators are physically independent, they all can be modelled by different ETS$_2$ site types. We also mentioned that this allows us to introduce a simplified site mapping (Definition 5.1) and site relabelling (Definition 5.2) schemes. Let T_a denote the set of the ETS$_2$ site types corresponding to the articulators shown in Table 5.1. The site type mapping

$$\text{TYPE}_a \colon SL_a \to T_a$$

Group	Organs	Group Size	Measurement Type
bilabial closure	UL, LL	6	EMA
tongue dorsum height	TD	4	EMA
tongue tip height	TT	4	EMA
labiodental contact	UI, LL	4	EMA
velic aperture	VL	4	EMA
velar contact	TD, VL	2	EPG
alveolar contact	TT, AR	2	EPG
palatal contact	TT, HP	2	EPG
voicing	VF	2	laryngeal

Table 5.2: Various groups of primitive gestures shown along with the crucial articulators (Table 5.1) participating in their formation, group sizes and the sources of available measurements.

corresponding to our representation is defined as a one-to-one (identity) mapping

$$\mathrm{TYPE}_a : T_a \to \mathbb{N}\,, \quad \text{where} \quad \forall t_i \in T_a \;\; \mathrm{TYPE}_a(t_i) = i\,,\; 1 \leq i \leq |T_a|\,.$$

Since there is a one-to-one correspondency between the site types and the site labels in our representation, any site relabelling also has to satisfy this property. In this study, we are not using any site relabellings since our sites are fixed.

Table 5.2 shows the groups of primitive gestures used in this study. For each group, the relevant sites (articulators), the number of distinct constituent gestures (primitives) and the sources of available measurements are shown. As was mentioned above, primitives, sites and primitive groups have been specified using the expert knowledge. Informally, a group consists of closely semantically *and* syntactically related primitive gestures involving similar articulators.

Example 5.9 (Articulatory Group of Gestures)**.** The group specifying the velic aperture consists of four gestures which correspond to the EMA trajectory of the velum, which is the only site these four primitives have. On the other hand, the velic closure group consists of syntactically different primitives which are derived from the EPG data and involve two articulators which correspond to the velum *and* the tongue back. As we shall see in the later sections of this chapter, the concept of an articulatory group allows us to introduce some domain-specific knowledge into the ETS_2 framework. ▷

Some of the groups of the ETS_2 primitives are presented in Figure 5.14. The four groups shown are:

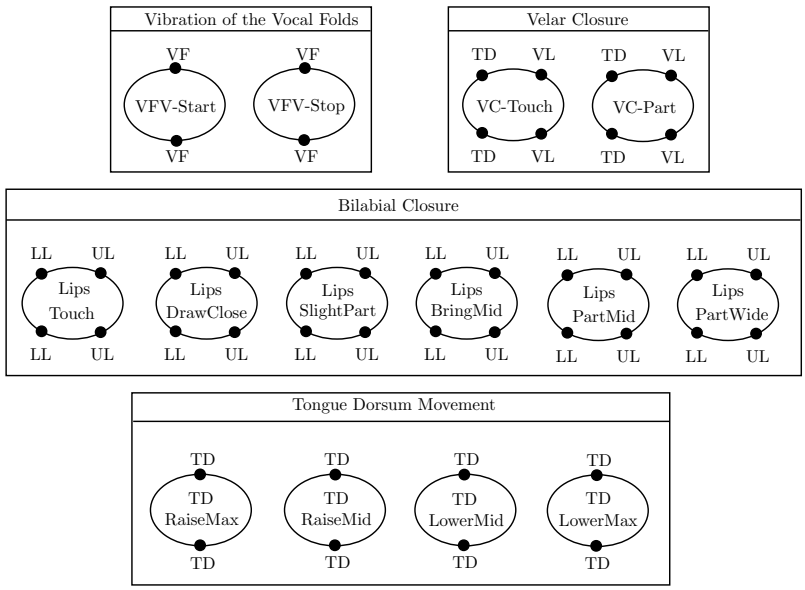


Figure 5.14: Some of the groups of primitive gestures from Table 5.2.

- *Vibration of the vocal folds:* This group consists of two primitives which describe the vibration of the vocal folds, in a binary fashion (on/off).
- *Velic aperture:* Similar to the above, this group also consists of two primitives which specify whether the velic closure has been detected.
- *Bilabial closure:* This group consists of six gestures which describe the movement of the upper and lower lips. Note that the these gestures model the trajectory and also distinguish between two different directions of movement (lips getting closer and lips parting).
- *Tongue dorsum height:* This group describes the vertical trajectory of the back of the tongue (its height) by four gestures. Similar to the gestures describing the bilabial states, these gestures also distinguish between two possible directions of the movement.

5.3.4 Automatic Detection of Primitive Gestures

Given an articulator (or group of articulators) of interest and the various corresponding streams of measurements, various groups of gestures can be detected. Below, we describe a simple pre-processor front-end for automatic detection of the primitive gestures in the data.

Vibration of the vocal folds (VF) that uniquely defines voiced and unvoiced sound patterns is represented by the two primitives standing for the beginning (VFV-Start)

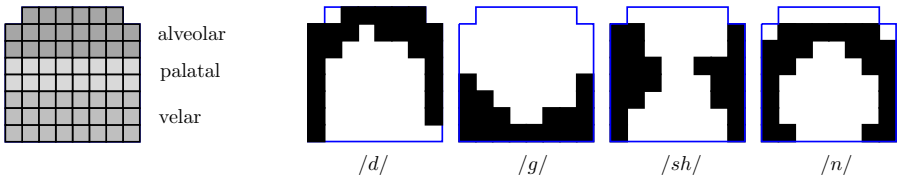


Figure 5.15: Three EPG regions and depiction of typical stable phases of the four consonants [d], [g], [sh] and [n] (after Figure 1 in Carreira-Perpiñán and Renals, 1998).

and end (VFV-Stop) of vibration respectively. The pitch detection algorithm used on the acoustic recordings provided by the MOCHA database is described by Talkin (1995). We used a 5 ms interval for analysis frames and a pitch frequency search range between 25 Hz and 600 Hz. Given the acoustic stream, at any given point in time the decision about the beginning and termination of the vibration is made when:

1. a change in the state of pitch is detected by the pitch detection algorithm, *and*
2. this new state is steady for at least 20 ms (around 320 samples of a 16 kHz recording), which is an minimum duration of a typical short vowel.

Given the EPG stream provided by MOCHA, it is possible to detect various contacts between the tongue and the hard palate. The output of the EPG sensor consists of 8 8-bit binary vectors with a simple spatial structure. The first three rows represent the alveolar region (the first and the last bit of the first row are unused), followed by two rows representing the palatal region, with the last three rows roughly corresponding to the velar region. Figure 5.15 illustrates the three main EPG regions and their typical behaviour during the articulation of the four consonants [d], [g], [sh] and [n].

In order to determine whether a contact has occurred, for each of the three regions (velar, palatal and alveolar) we use the *contact index* measured by the linear combination of the rows representing that region (which is a sum of all the bits of the rows), as described by Nguyen (2000). Given an appropriate per-region threshold (τ_a,τ_p and τ_v representing the alveolar, palatal and velar regions, respectively) defined by examining the relevant EPG measurements, change in the contact information at any given point results in the emergence of an appropriate primitive if and only if the threshold value of the index is crossed. For instance, the velar contact gesture VC-Touch emerges when the value of the velar index increases beyond τ_v, while the gesture VC-Part signifying the release of the closure emerges when this value decreases below τ_v. The emerging primitive gestures involve the pair of organs corresponding to the contact location. For palatal contact, the organs would involve tongue tip (TT) and the hard palate (HP), for alveolar contact the pair would include the tongue tip (TT) and the alveolar ridge (AR). Since the EPG sampling frequency of 200 Hz is reasonably low and the measurements

appear to change slowly over time, we have not imposed any requirements on the values of the indexes to be steady for any period of time.

The data stream containing EMA trajectories provides additional information about the articulations. Since the primitive gestures to be detected in the EMA data have a discrete nature, an obvious approach we follow is to cluster the distance measurements between the pair of the articulators of interest. The clustering procedure, making use of an efficient variant of k-means described by Kanungo *et al.* (2002), is applied to the entire data available for the particular speaker. Since vocal tract configurations vary from speaker to speaker, the clustering procedure is speaker-dependent. Each of the n cluster centroids represents one of the n discretised distances between the two articulators. For any given EMA frame, the distance between the two articulators is calculated and compared to the nearest cluster centroid. If the nearest centroid for this pair of articulators has changed since the last frame and the current articulation is sustained for at least m frames, the decision is made to fire a primitive which represents the event responsible for a change in the state of the articulation. We consider the articulation to be *sustained for m frames* if the measurements of the distances between the two articulators for each of the m frames fall into the same cluster.

If a single articulator is involved in a gesture (for instance, the gesture TT-LowerMid only involves one articulator), the height of the articulator is calculated according to $A_y - BN_y$, where A_y stands for the y coordinate of the articulator in question and BN_y for the y coordinate of the bridge of the nose (origin). Whenever two gestures are involved (for instance, any lip aperture gestures), the distance is calculated as the distance between their respective vertical coordinates.

Note that two distinct primitives are used to indicate the articulator entering and leaving the current quantisation region (cluster). For example, if we consider the medium range of the tongue dorsum heights, when the new cluster centroid represents a higher range, we represent this transition by the TD-RaiseMid gesture. Otherwise, if the new cluster centroid represents the lower range, the transition is represented by a different gesture TD-LowerMid. This behaviour is illustrated in Figure 5.16. The six transitions indicated on the right-hand side of the figure correspond to four primitive gestures since TD-LowerMax is identical to TD-RaiseMid and TD-RaiseMin is identical to TD-LowerMid.

5.3.5 Gestural Formations as ETS$_2$ Structs

In Section 5.2.2 we mentioned that an ETS$_2$ *struct* is a temporally ordered sequence of connected primitives capturing the "history" of the corresponding process. Within an articulatory representation, a struct is identified with a temporal sequence of primitive

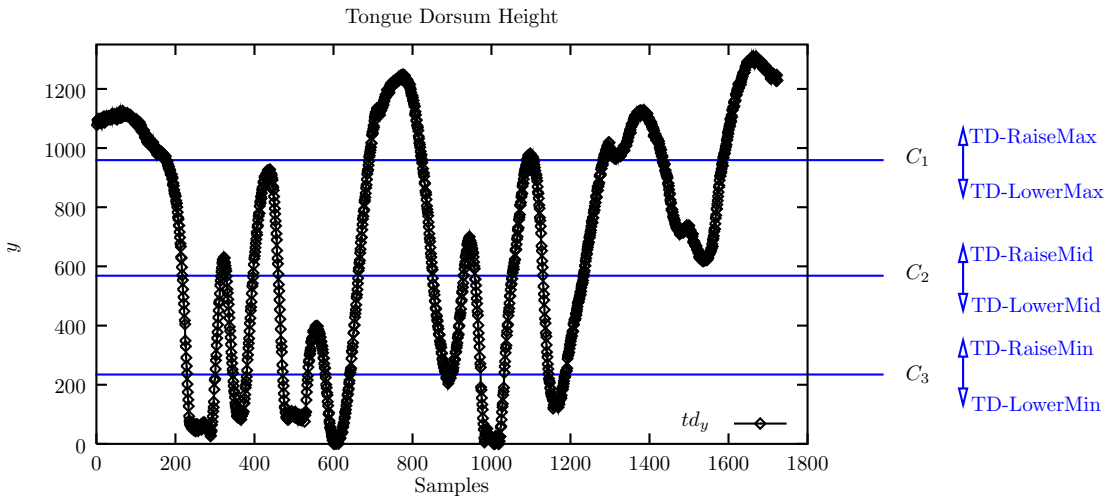


Figure 5.16: Detection of four distinct tongue dorsum height gestures in the corresponding EMA stream. Six states correspond to four gestures. The direction of the movement is significant, hence TD-LowerMax is different from TD-RaiseMid and TD-RaiseMin is different from TD-LowerMid.

gestures, which are hypothesised to provide the gestural structure of any given utterance. We note that any utterance can itself be interpreted as a highly non-trivial gesture.

The inductive construction procedure of the ETS_2 structs, outlined in Definition 5.4, involves the attachment of the currently observed primitive transformation to the accumulated struct. For our representation, this mechanism may be suboptimal. In reality, at any given point in time the pre-processing algorithm (responsible for the derivation of the gestural structure from the real data) may observe several instances of different primitive gestures appearing simultaneously. In order to formally allow this, the ETS_2 model has to be modified to allow for partial order of the primitives in the structs, since currently it does not support this. A domain-specific extension, however, is possible. The following *gestural structure construction* procedure allows us to support simultaneous articulatory events at least partially:

1. If at any point in time, a single primitive is observed, the struct is grown as usual using the procedure outlined in Definition 5.4.

2. Suppose several primitives that do not share any articulators in common are observed simultaneously. Since these primitives do not share any sites, no confusion arises and these primitives are represented as an unordered tuple.

3. Otherwise, let k be the number of simultaneously observed primitive gestures which share some sites in common. In addition, let t_i be the time of the current speech frame being processed. The k primitives are "de-parallelised" in such a

way that all of them appear sequentially *before* the next speech frame starts at t_{i+1}. During this serialisation of primitive gestures, the regular ETS$_2$ struct construction is employed since primitives are now added one by one.

In practice, however, the steps (2) and (3) of the above procedure are seldom employed because the simultaneous appearance of primitive articulatory events is quite rare. We employ the above construction procedure for the derivation of the gestural structures in form of ETS$_2$ structs for any given utterance.

Example 5.10. Figure 5.17 shows an ETS$_2$-based gestural structure of the word "get", consisting of 11 primitive gestures operating on 5 articulators, together with the corresponding phonetic segments, which are shown purely for convenience (detection and construction processes do not make use of these segments). Names of all the articulators (corresponding to ETS$_2$ site types) are given in Table 5.1. The gestural structure in Figure 5.17 is constructed on-the-fly from the primitive gestures detected in the available articulatory and acoustic data. For the sake of clarity, only some of the primitive gestures participating in the critical articulation of the voiced velar stop [g] and the unvoiced alveolar stop [t] are shown.

The articulation of [g], for instance, has a simple interpretation within this representation. Articulation is achieved by first forming a velar constriction, which, in turn, is formed by the tongue dorsum TD first rising to its maximum position (TD-RaiseMax) at 0.248 sec, then completing the constriction before the phoneme boundary by touching the velum VL (VC-Touch) at 0.266 sec. The constriction is released within the phoneme boundaries of [e] by first slightly lowering the tongue dorsum TD (TD-LowerMid) at 0.416 sec and then parting the tongue dorsum TD from the velum VL (VC-Part) at 0.460 sec. Note that vibration of the vocal folds VF (VFV-Start) occurs at the onset of [g] at 0.380 sec. Similarly, it is possible to analyse the unvoiced alveolar stop [t], the articulation of which is obtained by means of the tongue tip (TT), alveolar ridge (AR), and the vocal folds (VF). ▷

5.3.6 Articulatory Transformations

In Section 5.2.4 we mentioned that an ETS$_2$ *transform* is an encapsulation of a regular temporal pattern of primitives, which is subdivided into two parts: the *context* and the *body*. The context of a transform identifies the place, within a given struct, in which the application of the body of the transform becomes legal, while the body is the "chunk" that extends the (previously constructed) struct (Definition 5.9).

By examining the ETS$_2$ gestural structs, generated by the preprocessing algorithm described previously, several structurally and semantically related gestural fragments of

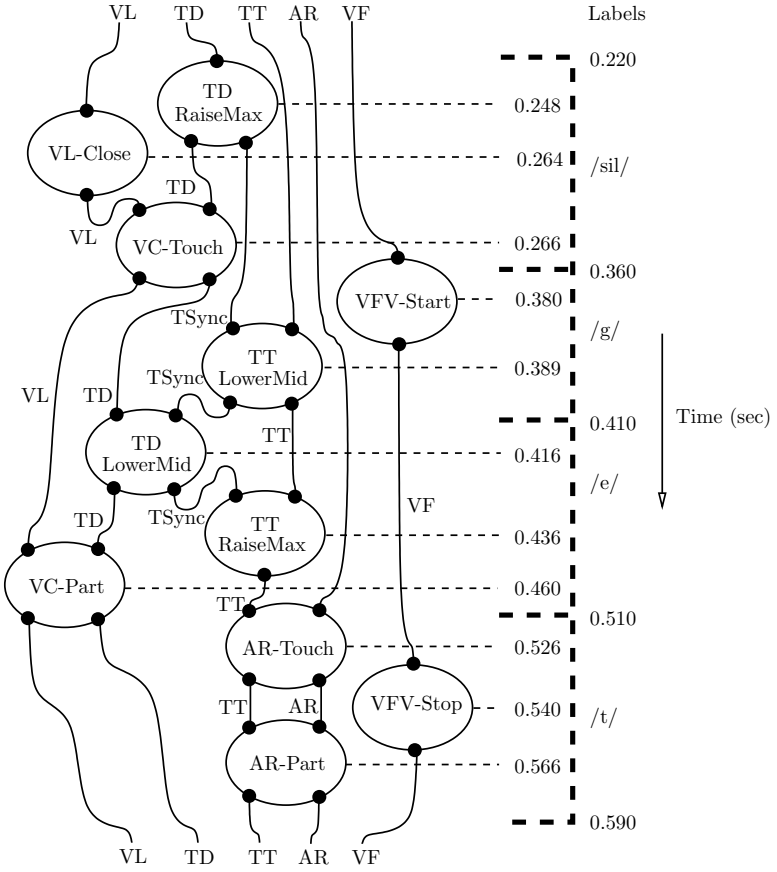


Figure 5.17: ETS$_2$ struct describing the gestural structure of the word "get", constructed using the automatically detected primitive gestures. Corresponding phonetic labels are shown.

the structs can be discerned. For each of the sound patterns of the consonants under investigation, the corresponding gestural fragments can be roughly divided into two parts, the actual constriction and the release. As mentioned in the previous section, the primitives comprising the two parts of the corresponding gestural fragment exhibit asynchrony and often span multiple phone boundaries (the anticipatory movement toward the lip constriction target, for instance, might start relatively early, before the constriction is actually produced). Therefore, in our analysis, the constriction starts with the first primitive gesture aimed at producing this constriction, ending with the last primitive gesture which secures its release.

Figure 5.18 shows some of the common (simplified) gestural patterns, four per sound, encountered in the data for unvoiced velar (top) and bilabial (bottom) stops [k] and [p]. The body of each of the ETS$_2$ transformations consists of the sequence of gestures which participate in the release of the stop, while the gestures which participate in the formation of the actual constriction are depicted as part of the transformation context. The context of the transformation can thus be seen as a necessary precondition for the respective sound to be produced (the gestures which are not critical for a particular articulation are shown with the connections to them crossed out).

Note that while each of the transformations has a similar higher-level semantics (for instance, all four transformations shown in the top figure represent the release of an unvoiced velar stop), structurally they are all different. The first two transformations differ in their bodies which can be interpreted as follows: For the first body, the release is accomplished by first removing the tongue back (TD) from the velum (VL) (the position of the tongue in the oral cavity is still high) followed by the lowering of the tongue back (TD). For the body of a second transformation, the tongue back appears to be lowered (together with velum) and only then detached. For all of the transformations, the vocal folds may have already stopped vibrating, meeting the necessary but not sufficient requirement for the articulation (in which case they are shown in the contexts) or, they stop vibrating at the onset of the release of that particular stop (in which case they are in the bodies of the transformations).

5.3.7 Class Description via ETS$_2$ Transforms

In Section 5.2.4 we mentioned that an ETS$_2$ *supertransform* is a set of closely-related transforms specifying the description of a class, where structural variations account for noise in the class (Definition 5.10). In the articulatory representation, a supertransform is identified with the family of temporal patterns of articulatory gestures (given by a family of transformations from Section 5.3.6) that collectively describe the class structure of a single phoneme.

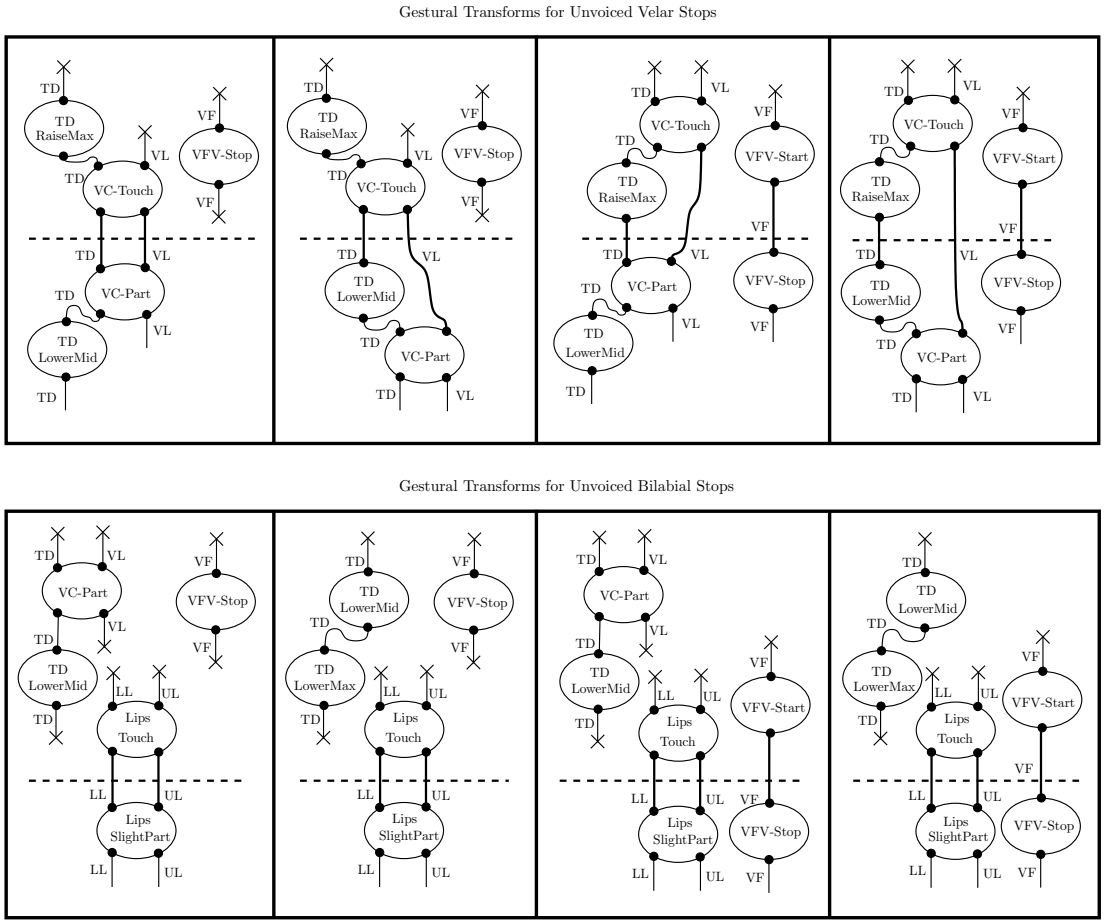


Figure 5.18: Simplified visualisation of some common gestural patterns encountered in the data for unvoiced velar (top) and bilabial (bottom) stops [k] and [p], represented as ETS$_2$ transformations.

Example 5.11. A simple class structure for the class of voiced velar stops defining phoneme [g] is given in Figure 5.19 (phoneme labels and timestamps corresponding to primitives are not shown). Each column of the supertransform representation consists of transforms which have structurally identical bodies (with each body specifying a release of constriction). The thicker lines connecting constituent gestures between the body and context of each constituent transform denote interface sites — used to indicate that the necessary precondition (provided by the context) for the articulation of the respective phoneme has been met. ▷

In Definition 5.10 of ETS$_2$ supertransform, it was mentioned that there is no restriction (apart from the one potentially introduced by the learning algorithm, described elsewhere by Goldfarb *et al.*, 2004, Part III) on the number of the primitives in either contexts or bodies of the constituent transforms. Therefore, one class supertransform can encapsulate the descriptions of various instances of gestural formations with different durations.

In this work, we have chosen to focus on class elements which are provided in terms of contexts of the gestural transforms. Our reason for doing this is that, for any given gestural transform, the detection of the context alone is enough to decide whether the phoneme corresponding to that transform has occurred. In addition, because phonetic labels have been provided with the data in this study, there is no need for duration modelling (for which one needs both body and context information). In other words, at present, we are not interested in the information indicating where each phoneme ends. This information is provided by the body of the transforms (to be more precise, by the last primitive gesture in transform's body). Henceforth, when referring to the gestural structure of phonemes, constituent gestural transforms are assumed to consist of contexts only.

Based on linguistic evidence (Ladefoged, 2001), only some of the primitive gestures from the gestural groups given in Table 5.2 were postulated to be critical for structural description of each of the 14 consonantal phonemes evaluated in this study. These phonemes are shown in Table 5.3. Alongside each phoneme P, the frequency N of occurrence of the corresponding label in the MOCHA corpus (the number shown is the same for both the male and female datasets) and the hypothesised constituent gestures are shown. Because we are interested in the contexts only, constituent gestures in Table 5.3 describe formations of various constrictions involved in production of phonemes in question.

For example, based on Table 5.3, the gestural structure of the unvoiced alveolar stop [t], is specified by a supertransform having six distinct constituent transforms (not shown), each consisting of various combinations of the three gestures VFV-Stop,

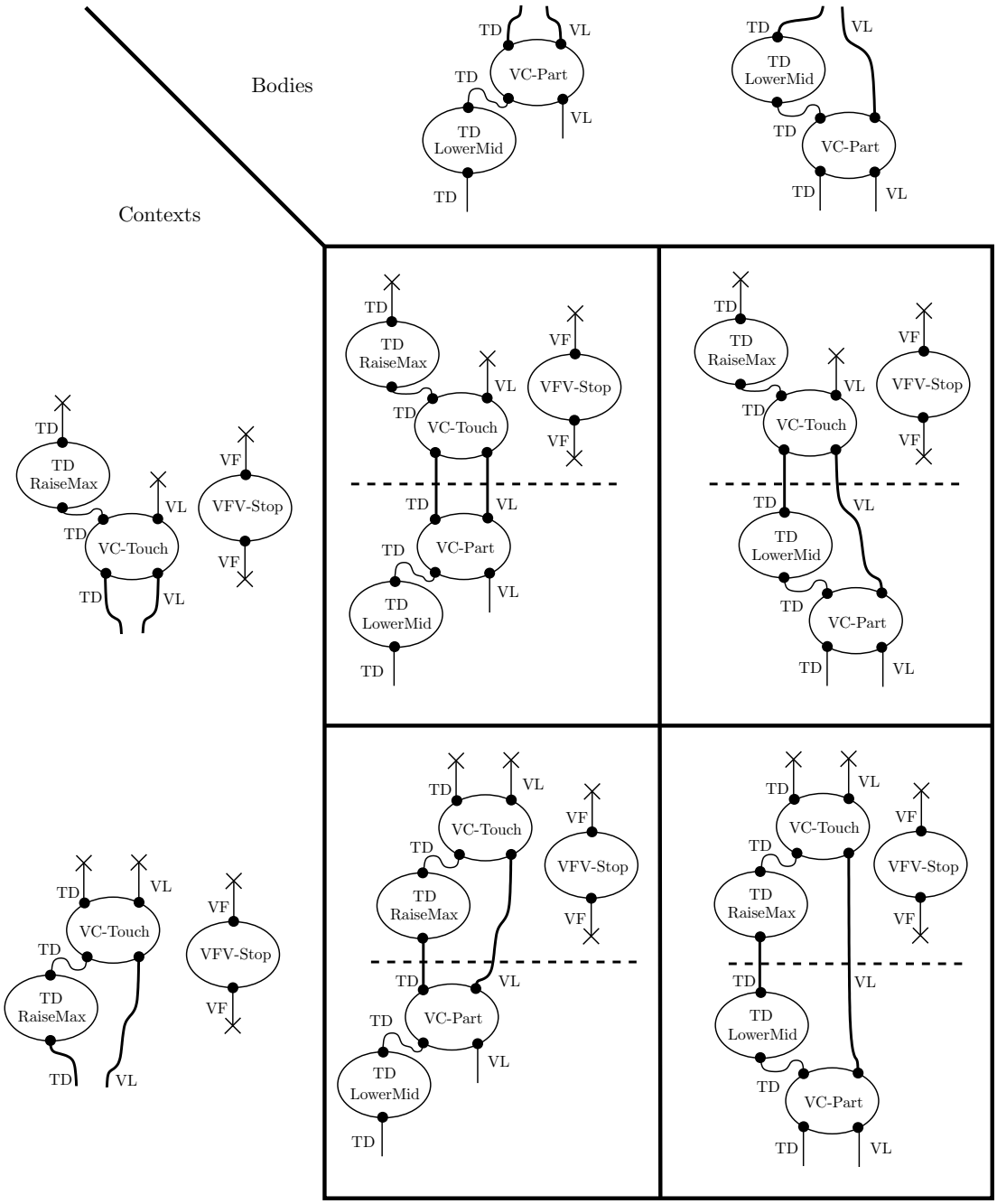


Figure 5.19: Simplified depiction of an ETS$_2$ supertransform for the class of voiced velar stops given by phoneme [g], consisting of four ETS$_2$ transforms.

P	N	Hypothesised Primitive Gestures
[b]	306	VFV-Start LipsTouch VC-Part AR-Part HP-Part
[p]	192	VFV-Stop LipsTouch VC-Part HP-Part
[g]	535	VFV-Start VC-Touch TD-RaiseMax AR-Part HP-Part
[k]	370	VFV-Stop VC-Touch TD-RaiseMax AR-Part HP-Part
[d]	531	VFV-Start AR-Touch TT-RaiseMax
[t]	871	VFV-Stop AR-Touch TT-RaiseMax
[v]	226	VFV-Start LD-Touch
[f]	263	VFV-Stop LD-Touch
[ng]	140	VFV-Start VC-Touch TD-RaiseMax VL-Close*
[m]	410	VFV-Start LipsTouch VL-Close*
[n]	835	VFV-Start AR-Touch TT-RaiseMax VL-Close*
[ch]	97	VFV-Stop TT-RaiseMax AR-Touch
[zh]	17	VFV-Start TT-RaiseMax HP-Touch
[sh]	146	VFV-Stop TT-RaiseMax HP-Touch

Table 5.3: Phonemes (P), the number N of corresponding per-speaker labels (examples) and the hypothesised constituent constriction-forming gestures (primitives) under investigation.

AR-Touch, and TT-RaiseMax.

5.3.8 Matching Gestural Transformations

Given an observed gestural formation, constructed using the inductive procedure outlined in Section 5.3.5, it is desirable to have a procedure for detecting the presence of an ETS_2 transform in ETS_2 struct corresponding to that formation. If an ETS_2 transform is located within a struct, this is an indication that a class (defined by a supertransform) to which this transform belongs, participates in the construction of an object represented by a struct.

As mentioned above, the pre-processing front-end detects primitive gestures in the available streams and employs the inductive construction procedure for updating the currently observed gestural formation (struct). For the detection of the transform, we only need to consider the case when the addition of a primitive to the end of a struct causes the "completion of construction" of a transform. Thus, the algorithmic approach presented here may be seen as a rooted depth-first search (Valiente, 2002), commencing with the last primitive in the body of a transform (corresponding to the latest primitive in the struct to be searched).

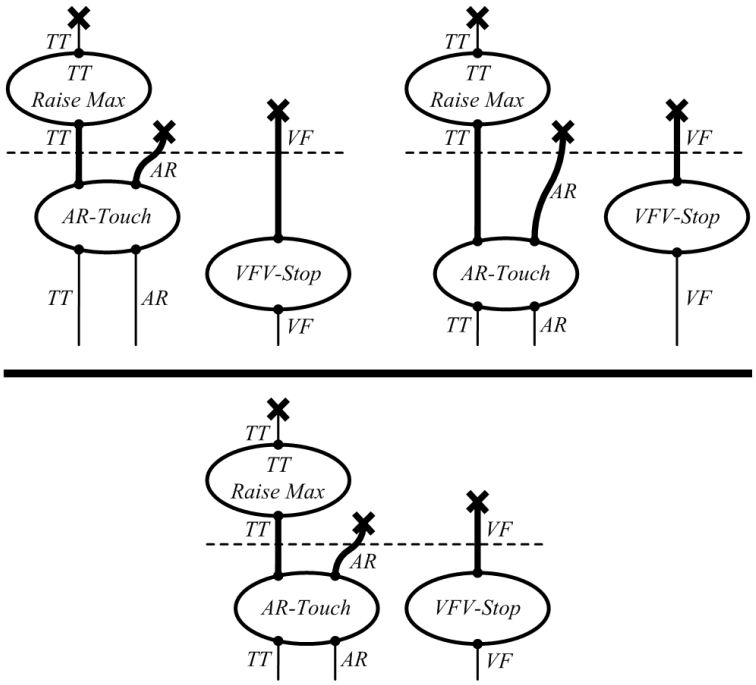


Figure 5.20: Top: Pictorial representation of two distinct transform fragments. Bottom: Representation of the above fragments as a single fragment, w.r.t. the partial order specification. Note that the "bottommost primitives" are AR-Touch and VFV-Stop (neither is attached to any succeeding primitive).

In order to reduce the number of structurally-equivalent transforms in a super-transform, a generalisation of the specification of transform context/bodies to partial orderings of primitives — rather than total orderings — is possible (see Figure 5.20). A pseudocode version of structural matching algorithm algorithm, which we previously reported in (Gutkin and Gay, 2005b), is presented in Figure 5.21.

A transform is accepted when the structure of the transform is detected inside the searched struct, i.e. when all primitives in Γ are mapped to primitives of the same type and "interconnectedness" in Π. A mismatch in type or interconnection causes the rejection of a transform. The worst-case complexity of this algorithm is $O(m^2 \log m)$, where m is the number of primitives in the transform to be matched.

In Section 5.3.3 it was mentioned that all of the primitives used in this representation are divided into various articulatory groups, each consisting of semantically and structurally related primitives. Additionally, all m constituent primitives $\gamma \in \Gamma$ belong to separate groups (see Table 5.3). This allows for the trivial modification of the matching procedure specified above, whereby any primitive gesture $\pi \in \Pi$ which does not belong to any of the m groups is skipped during the search (without aborting the search procedure). This modification allows the detection of candidate transforms in

1. Let Π denote the (ordered) set of n primitives in the struct to be matched and Γ denote the (partially ordered) set of m primitives in the transform to be matched.

2. For each "bottommost" primitive $\gamma_i \in \Gamma$ (see Figure 5.20) that is of the same type as π_n, $\pi_n \in \Pi$, perform the following search:

   Declarations

   ρ: a *current primitive* ($\rho \in \Gamma$)
   V: a set of *visited* γ*-primitives*
   P: a set of *pending* γ*-primitives*
   E: an *equivalent primitive* mapping $E\colon \Gamma \rightarrow \Pi$

   $V \leftarrow \varnothing$
   $P \leftarrow \{\gamma_i\}$
   $E \leftarrow \{\gamma_i \mapsto \pi_n\}$
   `WHILE` $P \neq \varnothing$,
     $\rho \leftarrow P.\mathrm{Pop}()$
     `IF` $\rho \in V$, `CONTINUE`
     `FOR` each primitive α attached to ρ,
       Let β be the corresponding primitive attached to $E(\rho)$
       `IF` $\alpha \in V$, `NEXT`
       `IF` $\mathrm{Type}(\alpha) \neq \mathrm{Type}(\beta)$
         Try next γ_i (return to step 2)
       `IF` $E(\alpha)$ exists `AND` $E(\alpha) \neq \beta$
         Try next γ_i (return to step 2)
       `IF` $E(\alpha)$ does not exist
         $E \leftarrow E \cup \{\alpha \mapsto \beta\}$
         $P.\mathrm{Push}(\alpha)$
     $V.\mathrm{Push}(\rho)$
   `HALT:ACCEPT`

3. `HALT:REJECT`

Figure 5.21: ETS$_2$ Transform Matching Algorithm.

those cases when a "structural overlap" of various class elements appears in the data. We used this modification of the search algorithm in our work.

5.3.9 Higher Levels of Representation

The class of gestures defined by a corresponding supertransform becomes the next-level (non-trivial) gesture in the representational hierarchy. In particular, the bodies of the constituent transform leading to this non-trivial primitive specify the various instances of this non-trivial gesture observed in the data. Figure 5.22 shows the emergence of the next-level non-trivial gestures (shown as the next-level ETS$_2$ primitives on the right-hand side of the figure). The unvoiced velar stop consonants from the initial

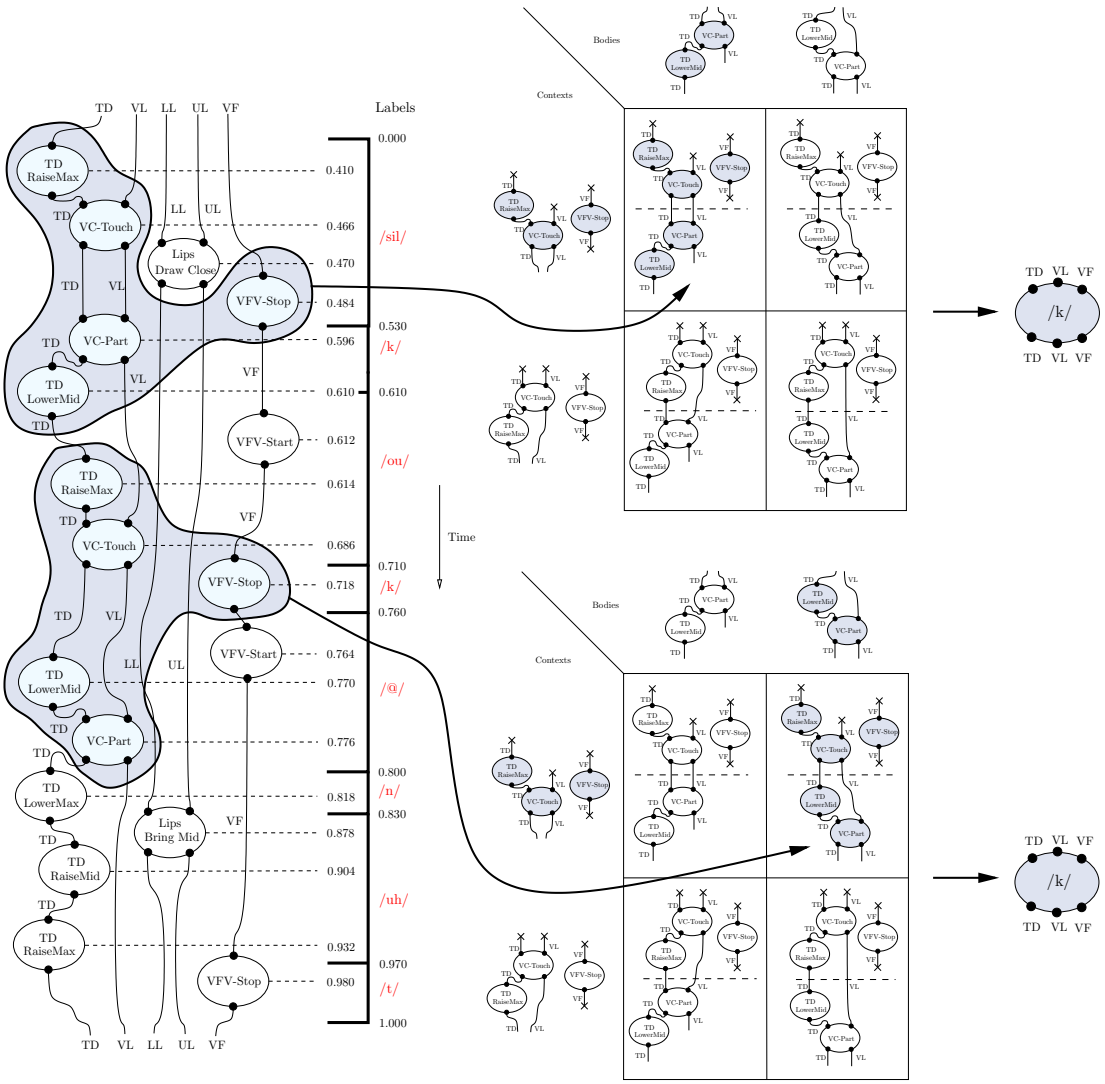


Figure 5.22: The two instances of the next-level primitive [k] emerging from a gestural structure of the word "coconut" via the supertransform for an unvoiced bilabial stop.

(articulatory) level gestural struct, corresponding to the word "coconut", are shown on the left-hand side. The shading shown indicates two different instances of initial-level gestural events which are represented at the next-level as the ETS$_2$ primitive [k]. Each of those instances corresponds to a different constituent transform from a class supertransform for [k] which is shown in the centre. The sites of a next-level primitive represent the organs participating in the formation of the class of events it represents. In the case of [k], the sites stand for the tongue dorsum, the velum and the vocal folds.

As mentioned in Section 5.2.6, once the new phonemic primitives "appear" on the next level, they can be used in the usual fashion in the construction of the next-level representation. Obviously, the primitives do not appear by themselves. Construction

of the representation at all the current levels is accomplished by the representation construction algorithm. Assuming that for each level we are given a fixed inductive structure (Definition 5.12), the new primitive appears on the next level if the representation construction algorithm can locate any of the constituent transformations of the associated supertransform in the struct at the current level. The search for constituent transform is performed using the gestural transform matching algorithm described in Section 5.3.8.

5.4 Experiments and Discussion

In this section we describe two sets of experiments aimed at verifying the adequacy of the proposed articulatory representation, based on the MOCHA articulatory corpus described in Section 5.3.2. The phonemic classes under investigation, corresponding to 14 consonantal phonemes of British English, are shown in Figure 5.3. Overall, there are 9,878 phonetic labels corresponding to the 14 phonemes in question, 4,939 per each speaker.

The first set of experiments, described in Section 5.4.1, focused on the verification of the primitive articulatory gestures automatically detected from the articulatory data using an algorithm described in Section 5.3.4. In Section 5.4.1, we provide the details of the verification algorithm, describe the parameters of the pre-processor responsible for detecting the gestures and provide the results.

The second set of experiments is described in Section 5.4.2. The experiments focused on the classification of the gestural class descriptions corresponding to the 14 consonantal phonemes under investigation. Each of the hypothesised gestural supertransformations (Section 5.3.7), consisting of the gestures shown in Table 5.3, were searched for within 920 (460 per speaker) ETS_2 articulatory formations automatically constructed from the utterances of the MOCHA corpus. The results of the search were then evaluated against the available phonetic labels. We describe the evaluation algorithm and discuss the results.

5.4.1 Gesture Detection

In order to evaluate the reliability of the primitive gestures described above, experiments were conducted to assess the potential accuracy of their detection. The evaluation was conducted on the `fsew` and `msak` data sets from the MOCHA corpus (Section 5.3.2). Since the corpus provides the phonetic labels, it is possible to check whether any of the primitive gestures, *a priori* known to participate in articulations which uniquely define certain phonemes, actually appear during runtime.

5.4.1.1 Verification Algorithm

The verification algorithm is applied to all the utterances in the corpus. For each phonetic label from a given utterance, each of the primitive gestures from a corresponding list is processed in turn. According to the algorithm, the primitive gesture *participates* in the formation of the corresponding phone if one of the following conditions is satisfied:

1. The primitive gesture appears within the boundaries (specified by the start and end times) of the phone label currently being processed.

2. The primitive gesture occurs somewhere within the boundaries of several previous phones.

In the second case, the algorithm checks that no other primitive gesture belonging to the same group occurred between the current phone and the phone where the primitive gesture of interest was detected. This is to ensure that the primitive gesture being verified (for example, LipsTouch) is not later cancelled by some other primitive gesture from the same group (for example, LipsSlightPart) before the current phone boundaries.

5.4.1.2 Experimental Setup

The list of 14 consonantal phonemes evaluated during the experiments is shown in Table 5.3. For each phoneme, the frequency of occurrence N of the corresponding label in the corpus is shown, along with the list of primitive gestures which are *a priori* hypothesised to participate in the formation of that phoneme. The frequencies of occurrence of the phonetic labels (4,939 labels in total) are equal for both male and female speaker data sets. Table 5.4 provides the description for each of the 13 critical gestures from Table 5.3. Each gesture is shown alongside the corresponding articulators it operates on, the data stream where the gesture is to be detected and a simple description. For example, the labio-dental closure LD-Touch involving the upper incisor and the lower lip is detected in the EMA stream. The name VL-Close* denotes a group consisting of *any* velic aperture gestures resulting in any degree of velum opening, excluding the closure.

The EPG parameters are $\tau_v = 12$, $\tau_p = 6$ and $\tau_a = 9$ for the velar, palatal and alveolar indexes, respectively. These values were determined by manually examining a small subset (two sentences, one for each speaker) of the corpus. The EMA steady state parameter m was set to 10 frames (20 ms for the EMA data sampled at 500 Hz). The number of EMA distance clusters n for all the pairs of articulators in questions was set to 3.

Gesture	Organs	Source	Semantics
LipsTouch	UL,LL	EMA	bilabial closure
VC-Touch	TD,VL	EPG	dorsum touches the velum
VC-Part	TD,VL	EPG	dorsum parts the velum
AR-Touch	TT,AR	EPG	alveolar closure
AR-Part	TT,AR	EPG	alveolar release
HP-Touch	TT,HP	EPG	palatal closure
HP-Part	TT,HP	EPG	palatal release
TD-RaiseMax	TD	EMA	raise dorsum high
TT-RaiseMax	TT	EMA	raise tongue tip high
LD-Touch	TT,UI	EMA	labio-dental closure
VL-Close*	VL	EMA	velum *not* closed
VFV-Start	VF	AC	vocal folds start vibrating
VFV-Stop	VF	AC	vocal folds stop vibrating

Table 5.4: Primitive gestures critical for the articulation of the phonemes given in Table 5.3

5.4.1.3 Verification Results

Validation experiments for each of the 13 critical gestures from Table 5.4 were conducted on the female and male data sets separately with the results shown in Table 5.5. Validation experiments employ the verification procedure defined above in Section 5.4.1.1. Because the original TIMIT phoneme labels are available, during this stage we know which gestures to anticipate. The error is calculated as the percentage of the primitive gestures which failed to meet the requirements of the verification procedure. For example, the number of expected occurrences (Ne) of gesture specifying the biliabial closure (LipsTouch) is 1086. The number corresponds to the number of bilabial closures which are *a priori* known to participate in makeup of various phonemes (such as [b] and [p]). The number of times this gesture has actually appeared during the validation of female speaker dataset (N_o^f) is 1078. Therefore, the accuracy for this gesture is 99.26% (1078 out of 1086) and the corresponding error (shown in Table 5.5 as E^f) is 0.74%.

The expected frequency of occurrence of each of the critical gestures N_e is the same for the male and the female speaker. For the female speaker, the observed frequency of occurrence of each gesture is specified by N_o^f and the error percentage is given by E^f. For the male speaker, the corresponding measurements are N_o^m and E^m, respectively. The overall error is 7.29% for the female speaker and 8.17% for the male speaker.

As can be seen from Table 5.5, while the overall error is reasonably low, some of the

Gesture	N_e	N_o^f	E^f (%)	N_o^m	E^m (%)
LipsTouch	1086	1078	0.74	1079	0.64
VC-Touch	867	803	7.38	750	13.49
VC-Part	676	670	0.89	644	4.73
AR-Touch	2334	2052	12.08	1870	19.88
AR-Part	727	716	1.52	727	0.00
HP-Touch	163	162	0.61	163	0.00
HP-Part	1403	1209	13.86	1325	5.56
TD-RaiseMax	867	854	1.50	844	2.65
TT-RaiseMax	2497	2352	5.81	2388	4.37
LD-Touch	489	479	2.04	481	1.64
VL-Close*	1385	1015	26.71	1086	21.59
VFV-Start	2657	2558	3.73	2573	3.16
VFV-Stop	2282	2213	3.02	2078	8.94
Total	17433	16161	**7.29**	16008	**8.17**

Table 5.5: Evaluation results for each of the primitive gestures for the female (`fsew`) and male (`msak`) speaker data sets.

primitive gestures are not detected very accurately. The problematic gestures are the alveolar contact (AR-Touch) between the tongue tip and the alveolar ridge (determined from the EPG data), the velar closure (VC-Touch) formed by the tongue dorsum and the velum (determined from EPG data) and the group of gestures (VL-Close*) defining the nasals (detected in EMA data). The latter inaccuracy in the detection of the nasalisation from the EMA data has been observed by others (Richmond, 1999). It is hypothesised that the EPG and EMA detection errors are due to the recording setup, where the subjects get used to the presence of the EMA coils and EPG palate and modify their articulation of the sounds in question, skipping some of the critical articulations. In addition, as observed by Richmond (2001), the sensors tend to dislocate during the recordings, causing inaccurate measurements.

5.4.2 Phoneme Classification

The aim of the experiments described below was to assess the performance of the structural identification of the 14 ETS supertransforms, each describing a consonantal phoneme of English,in gestural ETS_2 structures derived from real articulatory data. The algorithm for matching the articulatory transformations was presented in Sec-

tion 5.3.8. The reason for selecting this particular subset of 14 consonantal phonemes is simple. In this work we are primarily interested in sounds which are produced by various constrictions of the articulators (such as stops), because these sounds are clearly manifest in the articulatory data. Combining the articulatory and acoustic data for the recovery of vowels will be part of our future work, because other sounds, such as vowels, are more difficult to model (structurally) based on the articulatory evidence alone.

5.4.2.1 Evaluation Strategy

The evaluation was applied to all 920 gestural structures (460 per speaker) automatically derived from the utterances of the corpus. Overall, 9,879 phonetic labels were available for the 14 classes corresponding to the 14 ETS_2 supertransforms. In general, a supertransform (phoneme class), was considered to match if any of its constituent transforms matched the gestural structure corresponding to the label.

Since the representation is asynchronous and the articulation of stop consonants is *anticipatory* (Ladefoged, 2001), primitive gestures are not constrained to appear within the phoneme boundaries of any given label. For such anticipatory articulation, the primitive gestures forming constrictions usually appear before the beginning of the phonetic label, often spanning multiple phoneme boundaries. For instance, most of the gestures participating in the articulation of the voiced velar stop [g] shown in Figure 5.17 appear before the beginning of the corresponding phonetic label. The gesture VC-Touch completing the constriction occurs at 0.266 sec, 94 ms before the phoneme boundary.

Given the above, the search boundaries for any given constituent transform are not restricted to the boundaries of the phonetic label, but also include the boundaries of several previous phonemes. Phonetic boundaries are specified in terms of the start and end times of a particular label. In particular, for each phonetic label and a candidate class element (transform) to be matched, the sought structure is declared as a successful match if it is identified by the search algorithm presented in Section 5.3.8 (starting from the end time of the phoneme label and proceeding backward in time) and if one of the following conditions is satisfied:

1. The candidate class element is located within the phoneme boundaries of the phonetic label;

2. The candidate class element is found to be overlapping with the beginning of a phoneme label (i.e. the formation of the constriction is anticipatory, beginning before the start of a phoneme boundary).

5.4.2.2 Results and Discussion

The overall results of the verification of the 14 classes of consonantal phonemes are presented in Table 5.6 in the form of a confusion matrix. For each of the classes, the number of correct matches is shown on the diagonal in bold. The number of class phonemes which failed to classify in any of the available classes is shown under [X]. The number of expected phonemes is given by N_e, while N_o stands for the number of correctly matched phonemes. The accuracy of the structural matching, denoted C, is given in the last column. As can be seen from Table 5.6, out of 9,878 phonemes, 7,679 were classified correctly and 278 failed to match against any of the available classes. The overall accuracy is 77.74%.

Analysis of the per-class statistics shows that the lowest accuracy of 62.16% was obtained for the alveolar nasal [n] which was often confused with voiced alveolar stop [d]. This could be explained by the fact that the postulated class structures of these sounds (see Table 5.3), are not sufficiently discriminative, differing by only one gesture (the production of [n] is achieved in the presence of the velic opening). Therefore, due to a failure of the pre-processor to detect the corresponding change in the state of the velum, [n] is often classified as [d]. The relatively frequent misclassification of the class [m] as [b] can also be attributed to the same cause. In general, it is expected that performance should improve with a more accurate pre-processor and better discriminating phonemic class descriptions, especially in the obvious cases when misclassification is not due to noisy data or to errors in linguistic labelling of the corpus.

5.5 Summary and Potential Improvements

We presented a novel structural representation of speech, developed within the ETS_2 formalism. The representational unit chosen was the *gesture*, which is seen as the interaction of the various physiological organs involved in the act of speech production. We have proposed an intuitively simple methodology for detecting the gestures in the continuous speech and shown that the gestures can be extracted with a reasonably low error rate (7.29% and 8.17% error on two speaker data sets of 31 minutes each). In addition, we have described several domain-specific alterations of the formal machinery which were necessary to efficiently deal with the continuous articulatory data. These alterations included the added support for the concept of a group of primitive transformations, as well as extension of the inductive construction procedure to have some preliminary support for the partial (rather than total) ordering of the primitive transformations.

We described the 14 classes of English consonantal phonemes in terms of non-trivial

Classes	[b]	[p]	[g]	[k]	[d]	[t]	[v]	[f]	[ng]	[m]	[n]	[ch]	[zh]	[sh]	[X]	N_o/N_e	C (%)
[b]	**495**	0	2	0	1	0	4	81	0	13	0	14	0	0	2	495/612	80.89
[p]	33	**644**	2	0	1	0	0	41	0	0	0	18	0	0	1	644/740	87.03
[g]	6	0	**287**	9	2	0	3	17	6	1	0	16	2	7	28	287/384	72.14
[k]	11	0	22	**794**	13	0	2	33	12	1	0	77	5	27	73	794/1070	74.21
[d]	49	0	10	0	**821**	0	8	20	12	7	0	71	24	13	27	821/1062	77.31
[t]	66	0	31	4	64	**1401**	6	59	5	5	0	0	11	27	63	1401/1742	80.42
[v]	1	0	1	0	1	0	**431**	9	0	0	0	4	0	0	5	431/452	95.35
[f]	11	0	1	0	0	0	5	**503**	0	0	0	0	1	3	2	503/526	95.63
[ng]	11	0	21	0	15	0	5	1	**207**	0	0	0	1	0	19	207/280	73.93
[m]	176	0	3	0	8	0	25	3	0	**600**	0	0	0	0	5	600/820	73.17
[n]	142	0	32	0	277	0	23	0	17	17	**1038**	23	49	1	51	1038/1670	62.16
[ch]	0	0	5	1	0	0	0	15	3	0	0	**156**	1	9	0	156/194	80.41
[zh]	0	0	0	0	0	0	0	0	0	0	0	0	**34**	0	0	34/34	100.00
[sh]	0	0	0	0	1	0	0	0	0	0	0	1	0	**288**	2	288/292	98.63
All															278	7679/9878	77.74

Table 5.6: Phoneme classification results for female (`fsew`) and male (`msak`) speakers, shown as a confusion matrix.

combinations of articulatory gestures. A structural matching algorithm capable of detecting an instance of one of the above classes inside a gestural structure (corresponding to a speech utterance) was also presented. The performance of the proposed class descriptions on real data (the MOCHA corpus) was evaluated, yielding an overall matching accuracy of 77.74%. Our results support the hypothesis that a structural representation of articulatory speech allows adequate identification of the phonemic classes.

Despite a general agreement that the use of articulatory information is highly beneficial (on both linguistic and physiological grounds), progress in that direction has been limited. This state of affairs may be attributed to a poor understanding of how the various parts of a speech production system interact. We believe that the use of a representational formalism that supports the description of *structural* classes of articulatory processes (such as ETS$_2$) will benefit both speech science and the linguistic community. In particular, such a formalism will guide the modelling of these complex speech production mechanisms.

Potential Improvements

Following are some important research directions we are planning to pursue in order to improve the modelling power of the articulatory representation:

The pre-processor:

Here we list some ideas on how to improve the pre-processing front-end responsible for constructing the representation:

- As mentioned in Section 5.3.4, the EMA component of the pre-processor essentially employs k-means clustering (binning of the articulatory ranges) in order to detect the articulatory gestures. No physiologically-inspired modelling is used. We currently believe that this approach is suboptimal.

  The modelling of the EMA trajectories can be improved by using a physiological model of the vocal tract based on the estimated tracing of the shape of subject's hard palate (as in Jung, 1993, Section 3.2.1).

- Detection of articulatory gestures from the EPG data can be further improved by using something more sophisticated than simple thresholding we are currently employing. For example, a statistical approach along the lines of the one suggested by Carreira-Perpiñán and Renals (1998) could be used.

- Additional examination of the phonetic segmentation of MOCHA. Originally it was obtained by means of automatic alignment and some of the labelling errors

were corrected by hand (Wrench and Hardcastle, 2000). However, some of the errors may have been missed out.

- Introduction of voting may be needed in order to improve the accuracy. For example, the gestures involving the velum can be detected both in the EMA and the EPG data. Sometimes the gesture is detected in the EPG stream but not in the EMA stream due to the noisy EMA measurements. The decision based on the EPG stream will override the EMA one if for the last several frames the EMA data was agreeing with the EPG data.

Representation:

Improvements in the representation can be obtained by introducing the following changes:

- Support for a mixed-mode acoustic/articulatory representation. In order to properly model vowels, articulatory measurements are definitely not enough. Luckily, MOCHA contains acoustic measurements corresponding to the articulatory ones. This modification will involve introduction of new primitives into the representation, which are detected from either the acoustic or laryngeal waveforms.

- The above modification will allow us to expand the list of classes we can reliably model to include the vowels. It will also help in modelling the difficult phonetic classes, such as semivowels. At the initial stage, an attempt would be made to come up with the class description (via ETS$_2$ supertransforms) for those classes and evaluation of those class descriptions on the MOCHA database.

- Since the representation is based on asynchronously-detected articulatory gestures, one need not focus on phonemes only. A more interesting linguistic class are syllables, which are becoming more popular as the units of linguistic and speech recognition analysis (Wester, 2003). Once the above issues are addressed, the problem can be transferred into the syllabic domain effortlessly.

Learning:

The discussion of the ETS$_2$ learning algorithm has been omitted from this thesis (apart from being mentioned briefly in Section 5.2.6). The interested reader is referred to (Goldfarb *et al.*, 2004, Part III). This is because we decided to focus on representations first. The focus on representational aspects explains why we did not address *learning* of the class structures, postulating them using *a priori* knowledge instead.

The algorithm, some basics of which have been briefly mentioned in Section 5.2.6 on p. 174, has nevertheless been successfully implemented and tested on the gestural

structures derived from MOCHA. There are several problems which need to be rectified before we can present the alternative formulation of the speech recognition problem within ETS$_2$. Briefly, the problems are the following:

- Currently, the algorithm operates in an unsupervised mode. This means that we have no control over the classes which are being discovered. What we need is the ability to present the learning algorithm with the finite number of instances of each class. In particular, the supervised version of the algorithm must ensure that we discover exactly one multi-level inductive structure (MIS) per class. This is important because the representation is asynchronous and correlating the result of the unsupervised learning with the available segmentation is difficult.

- The algorithm has several (numeric) control parameters which are difficult to correlate with the number of levels to be discovered, as well as the average size of the transform and the number of constituent transforms in the discovered supertransform. The algorithm needs to be changed to allow direct control over the above configurations.

Chapter 6

Conclusions and Future Research

In this chapter we conclude the thesis by providing a summary of obtained results, the main contributions and the directions for future research. This chapter is organised as follows. A summary of the thesis is presented in Section 6.1. The main contributions, their significance and relevance to spoken language modelling are listed in Section 6.2. Several dimensions of difficulty we encountered are discussed in Section 6.3. We conclude this chapter in Section 6.4, where the main future research directions are presented.

6.1 Thesis Summary and Results

This thesis explored the issues involved in structural representation of spoken language. The variety of approaches pursued in this thesis correspond to the evolution of our ideas and understanding of structural modelling. In the first part of this thesis (Chapters 2–4), a structural similarity-based (or topological) approach to modelling was investigated. We summarise our findings in Section 6.1.1. In the second part of this thesis (Chapters 5), formal approach to structural representation was explored. This is summarised in Section 6.1.2.

6.1.1 Topological Approach

First, we proposed a linguistically well-motivated structural representation. The atomic units of representation are distinctive phonological features. The objects under investigation corresponded to phones. We postulated the structure of objects in terms of distinctive phonological features and designed a similarity measure which operates on these objects. The similarity measure, together with the set of object representations automatically derived from speech, comprises a symbolic metric space. The quality of the proposed similarity measure was evaluated on dataset reduction and phone classification tasks involving the TIMIT corpus of read speech (Garofolo *et al.*, 1993). To this

end, we utilised several clustering and classification algorithms previously reported in the structural pattern recognition literature. In addition, we proposed a new initialisation criterion for symbolic clustering. Using this novel criterion, the system performed as well as simple statistical models (monophone HMMs) on the same task.

Next, we explored the transition from the above symbolic space to the corresponding similarity-based pseudo-Euclidean vector space representation. The primary motivation behind this transition was to construct an equivalent (similarity-preserving) representation in a space where efficient visualisation, classification and learning machinery is available. Several procedures for the construction of isometric vector space representations using pseudo-Euclidean embeddings were investigated. We conducted several classification experiments using both simple (k nearest neighbours) and non-trivial vector space classifiers (neural networks and support vector machines). We introduced a novel step in the algorithm for dimensionality reduction of the isometric embedding which, in some cases, resulted in improved classification performance. In general, the results support the hypothesis that classifiers constructed on similarity-based vector space representations perform as well (or better) than classifiers in the original symbolic space on both well-separable small (three classes) and full (39 classes) classification tasks. Furthermore, this is an indication that, from the point of view of the similarity measure, no information is lost in the transition from symbolic to vector space representation.

We then explored the learning of class representations in the original symbolic space using the ETS_0 model. The main motivation behind this stage of the thesis was to propose the procedure for the discovery of linguistically meaningful structural makeup of the phonemic classes at hand. To this end, we utilised the previously reported ETS_0 algorithms which involve learning of a class-specific similarity measure. This measure is induced by non-trivial class-specific structural features discovered by the algorithm. Training and classification experiments were conducted on both small and full tasks. We found that the ETS_0 training procedure discovers linguistically interesting class descriptions. We also demonstrated that for the small task the new class-specific similarity measures often result in improved classification performance in comparison with the original symbolic and vector space algorithms. On a full task, however, we demonstrated a degradation in classification performance of the discovered similarity measure, which was most likely due to violation of metric axioms.

Thus, we achieved the first research objective of this thesis, stated in Section 1.4 (p. 43) — to develop a linguistically well-motivated structural representation for phonemic objects and classes and experimentally evaluate it. The main conclusion drawn from the first part of this thesis is that the similarity measure plays an absolutely crucial role in all of the above developments. The careful selection of features and objects

for structural representation is important. Together with a good similarity measure, which accurately reflects the morphology of the domain, one can use two *general* frameworks for representation: a similarity-based vector space representation and ETS_0. Both frameworks stress the importance of the similarity measure. In the first case, the representation is constructed solely on the basis of similarities. The similarity measure provides a mathematical structure for the resulting vector space. In the second case, the discovery of the structural make-up of classes at hand amounts to the discovery of a class-specific metric, the learning process being guided primarily by the evolving similarity measure. In this case, the similarity measure provides the missing link between the representation of objects and the representation of classes. Therefore we believe that any future approaches to structural representation of spoken language should focus on the design of accurate similarity measures. As we hope to have demonstrated in this part of the thesis, the existence of general frameworks for representation, both structural and numeric, does not necessarily guarantee interesting discoveries if the metric is not designed well. Another important conclusion we drew in this part of the thesis is that the similarity-based and ETS_0 approaches are not mutually exclusive. Each approach possesses important features lacking in the other (ETS_0 provides us with structural class descriptions, while similarity-based vector-space representation provides us with powerful visualisation, classification and learning techniques). Therefore, we believe that any metric-based structural representation should employ both techniques for modelling.

6.1.2 Formal Approach

The recent versions of the ETS model introduced a novel formal language that explicitly addresses the issue of object and class representation. It does this by providing a uniform set-theoretic framework which incorporates the formal machinery for linking these concepts. An important feature of the formalism is its event-based (or process-based, in the last variation) philosophy. This allows one to structurally express the dynamic nature of speech production and perception processes in a single mathematical language. Initial step in this direction have been undertaken in the second part of this thesis.

We adopted a production-based, articulatory, view of spoken language and proposed a novel ETS_2 representation based on speech production principles. The representation is primarily motivated by the combinatorial view of speech advocated by the theory of articulatory phonology. We described the basic units of ETS_2 representation — the articulatory gestures — and presented a conceptually simple pre-processing algorithm for the automatic acquisition of these units from the articulatory (and some acoustic) data. We developed the methodology for evaluating the quality of these units on a standard

corpus of articulatory recordings. We showed that the basic units of articulatory representation can be recovered from articulatory measurements with a reasonably low error rate. Next, we introduced the entire sensory-level representation based on the articulatory gestures and described several representation-specific assumptions which allowed us to construct a procedure for evaluating the structural class descriptions. Based on the observations of articulatory structures automatically recovered from the data and on the linguistic evidence, we postulated the class structure of several phonemic sounds. The quality of these class structures was experimentally verified using a structural matching algorithm developed for this purpose. Classification results support the hypothesis that the articulatory class descriptions are important and beneficial for the identification of phonemes. Thus, we have achieved the second research objective of this thesis stated in Section 1.4 — to design and experimentally evaluate formal representation of speech based on articulatory principles.

Perhaps the most important lesson learnt from the application described in the second part of this thesis is the recognition of importance of the concept of *representation*. At the beginning of this thesis, we mentioned that two fundamental stages in pattern recognition consist of representation and generalisation (Section 1.1.1). Representation of articulatory processes was studied within a framework in which the concept of an object/process is, for the first time, *formally* related to the concept of a class. Unlike the "conventional" approach pursued in the first part of this thesis, the formal approach allows one to evaluate the quality of object/process representation at a very early stage in the design of a speech representation framework. This is because, by choosing the atomic units of sensory level representation, it is possible, at this early stage, to use the language of ETS for construction of simple class representations. The better the sensory level units reflect important information present in the data, the better the quality of the resulting class representations. We have shown that it is possible to construct intuitively interesting sensory level representations directly from the articulatory and acoustic data using straightforward pre-processing techniques. The atomic units of representations thus extracted, namely the articulatory gestures, are simple. At the same time, we have shown that these units describe important features of the data at hand. Within the formal language of ETS, these units combine together to form non-trivial structures and structural class representations that are linguistically interesting.

6.2 Contributions of this Thesis

The key contributions of this work are briefly listed below:

1. Design of a structural pattern recognition system based on linguistic principles

(Chapter 2);

2. Clarification of the relationship between phonemic object and class representation via the similarity measure, experimentally achieved using the dissimilarity-based (Chapter 3) and ETS_0 (Chapter 4) frameworks; automatic acquisition of structural make-up of phonemic classes using the ETS_0 framework; application of dissimilarity-based and ETS_0 frameworks to speech representaton and classification;

3. Development of a structural approach to articulatory representation of spoken language within the ETS_2 formalism; design, automatic acquisition and evaluation of basic units of articulatory analysis; formal clarification of the nature of non-trivial articulatory structures, and articulatory classes in particular; the first known attempt to tackle a pattern recognition problem with formal versions of ETS (Chapter 5);

6.3 Open Issues

The summary provided in each chapter of this thesis briefly lists the problematic issues and potential improvements. Some of these issues are purely "technical" and can be reasonably well resolved by either modifying the involved algorithms or by employing alternative techniques developed by others. Other issues point at more fundamental problems, some of which are reviewed below.

6.3.1 Topological Approach

The approach taken in Chapters 2–4 was to work with distinctive phonological features as the basic units of structural representation. While, from a linguistic point of view, this choice appears to be well-founded, it is not clear whether this is a good choice from the point of view of acoustic modelling stage of speech recognition. The reason for this doubt is simple. Distinctive phonological features are too abstract to be associated with the signal directly. All of the distinctive phonological features have acoustic correlates. However, establishing these correlates automatically without recourse to some nonlinear mapping is difficult. Therefore, the pre-processor that we used for extracting structural atoms of representation makes use of the data produced by the non-trivial model employing recurrent neural networks (see Section 2.3.2 on p. 53) (symbolic atoms of structural representation are the quantised target values of distinctive feature-detecting neural networks). This model was used in (King and Taylor, 2000; King *et al.*, 2000; Wester, 2003) to extract distinctive feature estimates from speech. Thus, there is an intermedi-

ate stage between the acoustic signal and the structural pattern recognition framework. This intermediate stage is in itself a pattern recognition system that needs to be trained (in a supervised mode) in order to provide accurate estimates. One of the shortcomings of this approach is that the intermediate stage may introduce errors and inconsistencies into the structural representation, degrading its classification performance.

While the representation of objects (phones) within the structural framework is linguistically interesting, there are still several important issues which were not addressed. The phones are represented by objects called phonological templates. Each phonological template has knowledge of its own speech frames, but lacks any knowledge about the context, which in its simplest form is defined as the preceding phone (see Section 2.3.4 on p. 57). Each template consists of several strings (streams). This particular template structure prevents us from introducing any form of structural dependencies between the streams, apart from numerical ones. Modifying the object structure to account for both the contextual variation and inter-stream structural dependencies will result in a much more complicated (graph-like) structure. This, in turn, would require re-working of all the symbolic algorithms which are not purely based on the similarity measure, the template-based ETS_0 learning algorithm in particular. In addition, it is not clear how to structurally capture possible numeric correlations between various multi-valued features potentially present in the data.

In the experiments conducted in the first part of the thesis, we employed one of the most widely-used database of read speech — the TIMIT corpus. In statistical speech recognition, this corpus is often considered to be small. The spontaneous speech databases routinely used by large vocabulary continuous speech recognition (LVCSR) systems are more often than not considerably bigger. We discovered that even a reasonably small (in statistical speech recognition terms) corpus, such as TIMIT, poses several problems for the structural approach to modelling. The most problematic issue that we encountered is the issue of database pruning for construction of efficient similarity-based vector space representations. Initially, we constructed 124,962 symbolic object representations that correspond to a 39-class training portion of TIMIT. On average, this corresponds to 3,204 objects per class. In Chapter 2, we used symbolic clustering algorithms to reduce the size of this set. However, we also saw that the vector space embedding of the symbolic space is analytically more developed and, in particular, provides better developed clustering techniques. In fact, if one was able to construct a vector space embedding of 124,962 symbolic objects, no symbolic-space clustering would be needed at all because the structure of the vector space would allow for more computationally efficient storage and generalisation procedures. Unfortunately, it is computationally intractable to construct an embedding of the entire training set

because it is too large. The alternative which we followed was to reduce the size of the training set in the symbolic, rather than vector, space and only then construct a vector space embedding of the reduced training set. Thus, we were not able to resolve the issue of constructing pseudo-Euclidean space embeddings of large symbolic sets. We still have not found a feasible solution, although it is clear that such a solution will involve an incremental embedding.

Another issue is the reliability of the similarity measure. Is it possible to improve the performance of the system by using an alternative similarity measure which better reflects the structure of the objects? The answer to this question is affirmative. What is unclear is how to incorporate this measure into the ETS_0 learning process without breaking the optimisation. The Levenshtein-like form of the similarity measure which we used in the first part of this thesis allowed for a reasonably straightforward extension to a block-based edit distance used by the ETS_0 learning algorithm. The resulting distance, however, is not a metric. Worryingly enough (but hardly surprisingly), the semimetric or pseudo-metric properties of the similarity measure may affect the quality of the learning stage and result in the discovery of class descriptions which are not discriminating enough. We were not able to fully resolve this issue.

6.3.2 Formal Approach

In Section 1.3.4 on p. 37 we suggested an informal way of thinking about the speech communication process based on the multi-level ETS representation "tower". The receiver of the linguistic message reassembles it from acoustics by ascending the levels in a multi-level representation, possibly updating and growing it. Ascension to a new level is made possible due to detection, at each level of representation, of important perceptual features present in the stimulus. Articulation is a reverse process which can be seen as the process of "collapsing" the multi-level representation into a compact analog acoustic encoding.

Because we are primarily interested in representations for speech recognition, the question that may be asked is: to what extent does the purely articulatory representation of Chapter 5 contribute to our understanding of perceptual representations. If we adopt the above view of the speech communication process, it would appear that the articulatory approach does not quite fit our philosophy. At present, we do not have sufficient knowledge about the extent to which articulatory information is utilised during the decoding of the acoustic message. Hence, intuitively at least, an articulatory approach appears to be a very indirect way to look at perceptual mechanisms. To a certain extent the latter observation is in disagreement with the motor theory of speech perception (Liberman and Mattingly, 1985). Integration of more acoustic information

into the model may be need in order to make the representation more interesting and realistic. We nevertheless believe the articulatory representation may indeed provide several important clues which may not otherwise be discovered by the pure acoustic-based approaches.

An additional issue with the articulatory representation is that the very nature of the basic units of representation does not allow us to adopt a uniform approach to data: detection of each group of gestures requires the pre-processor to incorporate specific *a priori* knowledge about each type of articulator. In other words, each gesture requires the pre-processor to incorporate knowledge about the number of articulators involved in that gesture, measurement-specific detection components (e.g. EPG and laryngeal event detectors) and so on. Events from all these sources are incorporated into a single sensory level of representation. It is still not clear whether this task has been accomplished in a satisfactory manner. For instance, at present we are not aware of any structurally elegant ways of incorporating the events from articulatory and acoustic streams in a single level of representation. This would pose problems when we introduce additional acoustic events into the representation — at present, only the voicing events come from the acoustic stream of measurements. In addition, the articulatory representation is corpus-specific. It is very difficult to switch to another articulatory corpus which lacks some of the measurement sources we are currently using. In contrast, the purely auditory representation, if such was developed, could present a more uniform view of the data, primarily because the representation is constructed from a single source of measurements — an acoustic waveform. An additional important practical advantage of the auditory approach is that the acoustic data is available in larger quantities and is easier to acquire than its articulatory counterpart.

6.4 Future Work

In this section we outline some of the primary directions for future research.

6.4.1 Articulatory Representation

First of all, the articulatory representation needs to be constructed within the ETS_4 formalism. This will primarily involve reinterpreting the structure and function of the articulatory gestures within ETS_4. At present, it appears that very few changes are needed: a sensory level of the ETS_4 representation will be similar to our present ETS_2 interpretation.

The overall list of potential improvements which can be introduced into the articulatory representation has been provided in Section 5.5. The primary direction of research

will involve the development of a more sophisticated pre-processor for the acquisition of articulatory gestures. Better modelling of the physiological structures involved in the process of articulation will improve the quality of the sensory level of representation. At present, the pre-processor makes minimal use of geometry of the articulators and surrounding structures. Modelling of these structures is known to be beneficial to computational physiology (Engwall, 2003, 2004) and articulatory phonology (Jung, 1993; Jung *et al.*, 1996). An alternative articulatory corpus containing more articulatory measurement sources may need to be considered. We would be interested in obtaining accurate estimates of the gestures involved in production of vowel sounds. In particular, these will involve estimates of various tongue and labial configurations, which give important articulatory clues to the classification of vowels.

6.4.2 Speech Recognition Problem Revisited

An important direction for future research will involve work on a generalisation mechanism that could be used in a speech recognition setting. First, a supervised learning algorithm will need to be developed. Given a set of training examples belonging to some class, the goal of this stage is to discover the class representation. This problem is highly non-trivial. Ideally, we would like exactly one class (primal class, in ETS_4 terms or a supertransform, in ETS_2 terms) to emerge on the highest level of representation. In addition, each class should have the same number of pre-specified levels in its representation. This is because during the recognition stage, all the training set classes should ideally emerge on the same level. Otherwise, producing a transcription would be difficult because some of the classes may be found on different levels.

An additional issue involves the introduction of class-likelihoods into the model, which would allow multiple hypotheses to be produced during the recognition stage. This, in turn, will allow us to introduce a decoding component into the recogniser. The decoding component may be based on conventional principles and also allow the integration of a language model. At present, this seems to be a reasonable way of introducing *a priori* linguistic knowledge into the framework.

6.5 Concluding Remark

As noted by Deller *et al.* (1993), the observation made by James Flanagan in 1976 is a largely accurate reflection of the modern state of the field (Deller *et al.*, 1993, p. 604):

> "The problem of speech recognition has not been solved, primarily because the speech communication process is a subtle one. Many of its fundamentals are not very well understood. For example, while most researchers recognise that a short-time frequency spectrum of speech bears important

> information, the human ear and brain are not a laboratory spectrum analyser. We do not completely understand the inner ear, and what happens beyond an auditory nerve is almost a total mystery."

We believe that the formal study of representations of spoken language and the development of generalisation algorithms within these representations will eventually contribute to our understanding of the speech communication process. In this thesis we hope to have made a small, but important, step in this direction.

Bibliography

Abela, J. M. (**Jun. 2001**), "ETS Learning of Kernel Languages," Ph.D. thesis, Faculty of Computer Science, University of New Brunswick, Canada.

Abler, W. (**1989**), "On the particulate principle of self-diversifying systems," Journal of Social and Biological Structures **12**, 1–13.

Aiserman, M. A. (**1969**), "Remarks on two problems connected with pattern recognition," in *Methodologies of Pattern Recognition*, edited by S. Watanabe (Academic Press), p. 1.

Albert, A. (**1972**), *Regression and the Moore-Penrose Pseudoinverse* (Academic Press, New York).

Aldrich, J. (**1997**), "R.A. Fisher and the Making of Maximum Likelihood 1912–1922," Statistical Science **12**(3), 162–176.

Apostolico, A., Landau, G. M., and Skiena, S. (**Jun. 1998**), "Matching for Run-Length Encoded Strings," in *Proceedings of Complexity and Compression of Sequences'97* (IEEE Computer Society Press, Positano, Italy), pp. 348–356.

Arlazarov, V. L., Dinic, E. A., Kronrod, M. A., and Faradzev, I. A. (**1970**), "On Economic Construction of the Transitive Closure of a Directed Graph," Dokladii Akademii Nauk SSSR **194**, 487–488.

Arslan, A. N. and Egeciognu, Ö. (**2000**), "Efficient Algorithms for Normalised Edit Distance," Journal of Discrete Algorithms **1**(1), 3–20, special Issue on Matching Patterns.

Aubert, X. (**2000**), "A Brief Overview of Decoding Techniques for Large Vocabulary Continuous Speech Recognition," in *ISCA ITRW Automatic Speech Recognition: Challenges for the New Millenium* (Paris, France), pp. 91–96.

Bahl, L. R., Brown, P. F., de Souza, P. V., and Mercer, R. L. (**Jul. 1989**), "A tree-based language model for natural language speech recognition," IEEE Trans. Acoustics, Speech, and Signal Processing **37**(7), 1001–1008.

Bahl, L. R., Jelinek, F., and Mercer, R. L. (**1990**), "A maximum likelihood approach to continuous speech recognition," in *Readings in Speech Recognition*, edited by A. Waibel and K. Lee (Morgan Kaufmann, San Mateo, CA), chap. 6, pp. 308–319, reprint from *'83 IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Bicego, M., Murino, V., and Figueiredo, M. A. T. (**2004**), "Similarity-based classification of sequences using Hidden Markov Models," Pattern Recognition **37**(12), 2281–2291.

Bilmes, J. (**2003**), "Graphical Models and Automatic Speech Recognition," in *Mathematical Foundations of Speech and Language Processing*, edited by M. Johnson, S. P. Khudanpur, M. Ostendorf, and R. Rosenfeld (Springer-Verlag), vol. 138 of *The Institute of Mathematical Analysis Volumes in Mathematics and its Applications*.

Bird, S. and Liberman, M. (**2001**), "A formal framework for linguistic annotation," Speech Communication **33**(1,2), 23–60.

Bishop, C. M. (**1995**), *Neural Networks in Pattern Recognition* (Clarendon Press, Oxford).

Borg, I. and Groenen, P. (**1997**), *Modern Multidimensional Scaling* (Springer-Verlag, New York).

Bourlard, H. and Bengio, S. (**2002**), "Hidden Markov Models and other Finite State Automata for Sequence Processing," in *The Handbook of Brain Theory and Neural Networks*, edited by M. A. Arbib (MIT Press), 2 ed., pp. 528–533.

Bourlard, H., Hermansky, H., and Morgan, N. (**1996**), "Towards increasing speech recognition error rates," Speech Communication **18**(3), 205–231.

Brandenburg, F.-J. and Skodinis, K. (**2005**), "Finite graph automata for linear and boundary graph languages," Theor. Comput. Sci. **332**(1–3), 199–232.

Browman, C. and Goldstein, L. (**1989**), "Articulatory Gestures as Phonological Units," Phonology **6**, 201–251.

Browman, C. and Goldstein, L. (**1992**), "Articulatory Phonology: An Overview," Phonetica **49**, 155–180.

Bunke, H. and Csirik, J. (**1995**), "An improved algorithm for computing the edit distance of run-length coded strings," Information Processing Letters **54**(2), 93–96.

Bunke, H., Guenter, S., and Jiang, X. (**2001**), "Towards bridging the gap between statistical and structural pattern recognition: Two new concepts in graph matching,"

in *Proc. International Conference on Advances in Pattern Recognition (ICAPR'01)*, edited by S. Singh, N. Murshed, and W. Kropatsch (Springer-Verlag), vol. 2013 of *Lecture Notes in Computer Science*, pp. 1–11.

Bunke, H. and Jiang, X. (**2000**), "Graph Matching and Similarity," in *Intelligent Systems and Interfaces*, edited by H. N. Teodorescu, D. Mlynek, A. Kandel, and H. J. Zimmerman (Kluwer Academic Publishers), chap. 10, pp. 281–304.

Bunke, H. and Sanfeliu, A. (**1990**), *Syntactic and Structural Pattern Recognition - Theory and Applications*, vol. 7 of *World Scientific Series in Computer Science* (World Scientific).

Bunke, H. and Shearer, K. (**1998**), "A graph distance metric based on the maximal common subgraph," Pattern Recognition Letters **19**(3–4), 255–259.

Burges, C. J. C. (**1998**), "A Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and Knowledge Discovery **2**(2), 121–167.

Byrd, D. (**2003**), "Frontiers and challenges in Articulatory Phonology," in *International Congress of Phonetic Sciences (ICPhS)* (Barcelona, Spain).

Callahan, J. J. (**2000**), *The Geometry of Spacetime: An Introduction to Special and General Relativity*, Undergraduate Texts in Mathematics (Springer-Verlag, New York).

Carreira-Perpiñán, M. Á. and Renals, S. (**1998**), "Dimensionality Reduction of Electropalatographic Data Using Latent Variable Models," Speech Communication **26**(4), 259–282.

Casacuberta, F. and de Antonio, M. (**Apr. 1997**), "A Greedy Algorithm for Computing Approximate Median Strings," in *Proc. VII Spanish Symposium on Pattern Recognition and Image Analysis*, pp. 193–198.

Chomsky, N. (**1957**), *Syntactic Structures* (Mouton de Gruyter, The Hague).

Chomsky, N. and Halle, M. (**1968**), *The Sound Pattern of English* (MIT Press, Cambridge, MA).

Choueiter, G. and Glass, J. (**2005**), "A Wavelet and Filter Bank Framework For Phonetic Classification," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-05)*, vol. 1, pp. 933–936.

Clarkson, P. R. and Moreno, P. J. (**1999**), "On the use of Support Vector Machines for Phonetic Classification," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-99)*, vol. 2, pp. 585–588.

Clements, G. N. and Hume, E. V. (**1995**), "The Internal Organisation of Speech Sounds," in *The Handbook of Phonological Theory*, edited by J. A. Goldsmith (Blackwell, Cambridge), pp. 245–306.

Cole, R. and Hariharan, R. (**1998**), "Approximate string matching: A simpler faster algorithm," in *Proc. 9th Annual Symposium on Discrete Algorithms*, pp. 463–472.

Collobert, R. and Bengio, S. (**2000**), "Support Vector Machines for Large-Scale Regression Problems," Tech. Rep. IDIAP-RR 00-17, IDIAP, Martingy, Switzerland.

Collobert, R. and Bengio, S. (**2001**), "SVMTorch: Support Vector Machines for Large-Scale Regression Problems," Journal of Machine Learning Research **1**, 143–160.

Cortes, C. and Vapnik, V. (**1995**), "Support-Vector Networks," Machine Learning **20**, 273–297.

Davis, R. and Shrobe, H. (**1993**), "What is a Knowledge Representation?" AI Magazine **14**(1), 17–33.

de la Higuera, C. and Casacuberta, F. (**2000**), "The topology of strings: two NP-complete problems," Theoretical Computer Science **230**, 39–48.

Deller, J. R., Proakis, J. G., and Hansen, J. H. L. (**1993**), *Discrete-Time Processing of Speech Signals* (Macmillan).

Denecke, K. and Wismath, S. L. (**2002**), *Universal Algebra and Applications in Theoretical Computer Science* (Chapman & Hall/CRC).

Deng, L. (**1998**), "A dynamic, feature-based approach to the interface between phonology and phonetics for speech modeling and recognition," Speech Communication **24**, 299–323.

Deng, L. (**Nov. 2000**), "Spontaneous speech recognition using a statistical coarticulatory model for the vocal-tract-resonance dynamics," J. Acoust. Soc. Am. **108**(5), 1–13.

Deng, L., Ramsay, G., and Sun, D. (**1997**), "Production models as a structural basis for automatic speech recognition," Speech Communication **22**, 93–111.

Devroye, L., Györfi, L., and Lugosi, G. (**1996**), *A Probabilistic Theory of Pattern Recognition*, vol. 31 of *Stochastic Modelling and Applied Probability* (Springer-Verlag).

Dieudonné, J. (**1960**), *Foundations of Modern Analysis* (Academic Press, New York).

Droste, M., Pech, C., and Vögler, H. (**2005**), "A Kleene Theorem for Weighted Tree Automata," Theor. Comput. Sci. **38**(1), 1–38.

Duda, R. O., Hart, P. E., and Stork, D. G. (**2001**), *Pattern Classification* (John Wiley & Sons, New York), 2nd ed.

Duin, R. P. W. and Pękalska, E. (**2005**), "Open Issues in Pattern Recognition," in *Proc. Fourth International Conference on Computer Recognition Systems (CORES'05)*, edited by M. Kurzynski, E. Puchala, M. Wozniak, and A. Zolnierek (Springer-Verlag, Wrocław, Poland), Advances in Soft Computing, pp. 27–42.

Duin, R. P. W., Pękalska, E., Paclík, P., and Tax, D. M. J. (**Aug. 2004**), "The dissimilarity representation, a basis for domain based pattern recognition?" in *Pattern Representation and the Future of Pattern Recognition (Proc. Satellite Workshop of 17th International Conference on Pattern Recognition)*, edited by L. Goldfarb (Cambridge, UK), pp. 43–56.

Edelman, S. (**1998**), "Representation is Representation of Similarities," Behavioural and Brain Sciences **21**, 449–498.

Edelman, S. (**1999**), *Representation and Recognition in Vision* (MIT Press, Cambridge, MA).

Eden, M. (**1962**), "Handwriting and Pattern Recognition," IRE Transactions on Information Theory **8**, 160.

Ehrig, H., Engels, G., Kreowski, H. J., and Rozenberg, G. (**1999**), *Handbook of Graph Grammars and Computing by Graph Transformation* (World Scientific).

Eisner, J. (**2002**), "Parametric estimation for probabilistic finite state transducers," in *Proc. Annual Meeting of the Association for Computational Linguistics (ACL'02)* (Philadelphia, USA), pp. 1–8.

Engelfriet, J., Fülöp, Z., and Vögler, H. (**2002**), "Bottom-Up and Top-Down Tree Series Transformations," Journal of Automata, Languages and Combinatorics **7**(1), 11–70.

Engelking, R. (**1989**), *General Topology*, vol. 6 of *Sigma Series in Pure Mathematics* (Heldermann Verlag, Berlin), revised ed.

Engwall, O. (**2003**), "Combining MRI, EMA and EPG measurements in a three-dimensional tongue model," Speech Communication **41**, 303–329.

Engwall, O. (**2004**), "From real-time MRI to 3D tongue movements," in *Proc. 8th International Conference on Spoken Language Processing (ICSLP-2004)* (Jeju Island, Korea), vol. II, pp. 1109–1112.

Faundez-Zanuy, M., Bimbot, F., Davy, M., and Mori, R. D. (**2004**), "Special Issue on Non-Linear and Non-Conventional Speech Processing," Speech Communication **42**, 479–480, (call for papers).

Fischer, I. and Zell, A. (**2000**), "String averages and self-organizing maps for strings," in *Proc. 2nd ICSC Symposium on Neural Computation*, pp. 208–215.

Frankel, J. (**Apr. 2003**), "Linear dynamic models for automatic speech recognition," Ph.D. thesis, University of Edinburgh, UK.

Frankel, J. and King, S. (**Sep. 2005**), "A Hybrid ANN/DBN Approach to Articulatory Feature Recognition," in *Proc. 9th European Conference on Speech Communication and Technology (Eurospeech'2005)* (Lisbon, Portugal), pp. 3045–3048.

Fu, K. S. (**1982**), *Syntactic Pattern Recognition and Applications* (Prentice-Hall, New York).

Fülöp, Z. and Vögler, H. (**1998**), *Formal Models Based on Tree Transducers*, Monographs in Theoretical Computer Science (Springer-Verlag, New York).

Fülöp, Z. and Vögler, H. (**2004**), "Weighted Tree Transducers," Journal of Automata, Languages and Combinatorics **9**(1), 31–54.

Gantmacher, F. R. (**1959**), *Matrizenrechnung*, vol. I (VEB Deutscher Verlag der Wissenschaften, Berlin), translated from the 1st Soviet edition.

Garofolo, J. S. (**1988**), *Getting Started with the DARPA TIMIT CD-ROM: an Acoustic Phonetic Continuous Speech Database*, National Institute of Standards and Technology (NIST), Gaithersburgh, MD.

Garofolo, J. S., Lamel, L. F., Fisher, W. M., Fiscus, J. G., Pallett, D. S., and Dahlgren, N. L. (**Feb. 1993**), "DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus," Tech. Rep. NISTIR 4930, National Institute of Standards and Technology (NIST), Gaithersburgh, MD.

Gay, D. R. (**2005**), "ETS Representations in Computer Vision," Ph.D. thesis, Faculty of Computer Science, University of New Brunswick, Canada, in progress.

Gécseg, F. and Steinby, M. (**1984**), *Tree Automata* (Akadémiai Kiadó, Budapest).

Giegerich, H. J. (**1992**), *English phonology: An introduction*, Cambridge Textbooks in Linguistics (Cambridge University Press, Cambridge, UK).

Glass, J. (**2003**), "A Probabilistic Framework for Segment-Based Speech Recognition," Computer Speech and Language **17**, 137–152.

Goertzel, B. (**1993**), *The Structure of Intelligence: A New Mathematical Model of Mind*, Recent Research in Psychology (Springer-Verlag, New York).

Goldfarb, L. (**1979**), "Riemannian Representation of Finite Symmetric Spaces and Their Application to General Data Aanalysis," Ph.D. thesis, Department of Systems Design, University of Waterloo, Toronto, Canada.

Goldfarb, L. (**1984**), "A Unified Approach to Pattern Recognition," Pattern Recognition **17**(5), 575–582.

Goldfarb, L. (**1985**), "A New Approach to Pattern Recognition," in *Progress in Pattern Recognition*, edited by L. N. Kanal and A. Rosenfeld (Elsevier Science, Amsterdam), vol. 2, pp. 241–402.

Goldfarb, L. (**Jun. 1986**), "Metric Data Models and Associative Memories," in *Proc. 8th IASTED International Symposium on Robotics and Artificial Intelligence: Identification and Pattern Recognition* (Toulouse, France), vol. 3, pp. 53–73.

Goldfarb, L. (**1990**), "On the Foundations of Intelligent Processes – I. An evolving model for pattern learning," Pattern Recognition **23**(6), 595–616.

Goldfarb, L. (**1992**), "What is distance and why do we need the metric model for pattern learning?" Pattern Recognition **25**(4), 431–438.

Goldfarb, L. (**Aug. 2004**), "Representational Formalisms: Why we haven't had one," in *Pattern Representation and the Future of Pattern Recognition (Proc. Satellite Workshop of 17th International Conference on Pattern Recognition)*, edited by L. Goldfarb (Cambridge, UK), pp. 3–22.

Goldfarb, L., Abela, J., Bhasvar, V. C., and Kamat, V. N. (**1995**), "Can a vector space based learning model discover inductive class generalization in a symbolic environment?" Pattern Recognition Letters **16**(7), 719–726.

Goldfarb, L. and Deshpande, S. (**1997**), "What is a Symbolic Measumerement Process?" in *Proc. IEEE Conference on Systems, Man, and Cybernetics* (Orlando, FL), vol. 5, pp. 4139–4145.

Goldfarb, L., Deshpande, S. S., and Bhavsar, C. (**Apr. 1996**), "Inductive Theory of Vision," Tech. Rep. TR96-108, Faculty of Computer Science, University of New Brunswick, Canada.

Goldfarb, L., Gay, D., and Golubitsky, O. (**Sep. 2005**a), "What is a structural representation? Fourth Variation," Tech. Rep. TR05-174, Faculty of Computer Science, University of New Brunswick, Canada.

Goldfarb, L., Gay, D., and Golubitsky, O. (**Sep. 2005**b), "What is a structural representation? Fourth Variation," Pattern Recognition (under review).

Goldfarb, L., Gay, D., Golubitsky, O., and Korkin, D. (**Apr. 2004**), "What is a structural representation? Second Version," Tech. Rep. TR04-165, Faculty of Computer Science, University of New Brunswick, Canada.

Goldfarb, L. and Golubitsky, O. (**Oct. 2001**), "What is a structural measurement process?" Tech. Rep. TR01-147, Faculty of Computer Science, University of New Brunswick, Canada.

Goldfarb, L., Golubitsky, O., and Korkin, D. (**Dec. 2000**), "What is a structural representation?" Tech. Rep. TR00-137, Faculty of Computer Science, University of New Brunswick, Canada.

Goldfarb, L. and Nigam, S. (**1994**), "The Unified Learning Paradigm: A Foundation for AI," in *Artificial Intelligence and Neural Networks: Steps toward Principled Integration*, edited by V. Honavar and L. Uhr (Academic Press, Boston), pp. 533–559.

Goldstein, L. M. and Fowler, C. (**2003**), "Articulatory phonology: a phonology for public language use," in *Phonetics and Phonology in Language Comprehension and Production: Differences and Similarities*, edited by A. S. Meyer and N. O. Schiller (Mouton de Gruyter, Berlin), pp. 159–207.

Golub, G. H. and Loan, C. F. V. (**1983**), *Matrix Computations* (North Oxford Academic, Oxford).

Golubitsky, O. (**Mar. 2004**a), "On the Formalization of the Evolving Transformation System Model," Ph.D. thesis, Faculty of Computer Science, University of New Brunswick, Canada.

Golubitsky, O. (**Aug. 2004**b), "Turing-completeness of additive transformations in ETS," in *Pattern Representation and the Future of Pattern Recognition (Proc. Satellite Workshop of 17th International Conference on Pattern Recognition)*, edited by L. Goldfarb (Cambridge, UK), pp. 77–89.

Graepel, T., Herbrich, R., Bollmann-Sdorra, P., and Obermayer, K. (**1999**), "Classification on Pairwise Proximity Data," in *Advances in Neural Information Processing Systems (NIPS'12)* (MIT Press, Denver, USA), pp. 438–444.

Greenberg, S. (**Sep. 2001**), "Whither speech technology? A twenty-first century perspective," in *Proc. 7th European Conference on Speech Communication and Technology (Eurospeech-2001)* (Aalborg, Denmark), pp. 3–6.

Greenberg, S., Carvey, H., Hitchcock, L., and Chang, S. (**Mar. 2002**), "Beyond the Phoneme: The Juncture-Accent Model of Spoken Language," in *Proc. Human Language Technology Conference (HTL'02)* (San Diego, USA).

Greub, W. H. (**1967**), *Linear Algebra*, vol. 97 of *Die Grundlehren der mathematischen Wissenschaften* (Springer-Verlag, Berlin), 3rd ed.

Gusfield, D. (**1997**), *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology* (Cambridge University Press, New York).

Gutkin, A. (**2000**), "Log-Linear Interpolation of Language Models," Master's thesis, Department of Engineering, University of Cambridge, UK.

Gutkin, A., Gay, D., Goldfarb, L., and Wester, M. (**Aug. 2004**), "On the Articulatory Representation of Speech within the Evolving Transformation System Formalism," in *Pattern Representation and the Future of Pattern Recognition (Proc. Satellite Workshop of 17th International Conference on Pattern Recognition)*, edited by L. Goldfarb (Cambridge, UK), pp. 57–76.

Gutkin, A. and Gay, D. R. (**Sep. 2005**a), "A Formal Evolving Transformation System Approach to Structural Representation and Matching of Articulatory Speech Structures," in *Proc. 9th European Conference on Speech Communication and Technology (Eurospeech'2005)* (Lisbon, Portugal), to appear. Short version of Gutkin and Gay (2005b).

Gutkin, A. and Gay, D. R. (**May 18–20 2005**b), "Structural Representation and Matching of Articulatory Speech Structures based on the Evolving Transformation System (ETS) Formalism," in *Proc. 19th International Workshop on Qualitative Reasoning (QR-05)*, edited by M. Hofbaur, B. Rinner, and F. Wotawa (Graz, Austria), pp. 89–96.

Gutkin, A. and Gay, D. R. (**Aug. 2005**c), "Structural Representation and Matching of Articulatory Speech Structures based on the Evolving Transformation System (ETS) Formalism," in *Proc. 19th International Joint Conference on Artificial Intelligence*

(IJCAI-05), edited by L. P. Kaelbling and A. Saffiotti (Edinburgh, UK), pp. 1684–1685, short version of Gutkin and Gay (2005b).

Gutkin, A. and King, S. (**Oct. 2004**a), "Phone Classification in Pseudo-Euclidean Vector Spaces," in *Proc. 8th International Conference on Spoken Language Processing (ICSLP-2004)* (Jeju Island, Korea), vol. II, pp. 1453–1457.

Gutkin, A. and King, S. (**Aug. 2004**b), "Structural Representation of Speech for Phonetic Classification," in *Proc. 17th International Conference on Pattern Recognition (ICPR'04)* (IEEE Computer Society Press, Cambridge, UK), vol. 3, pp. 438–441.

Gutkin, A. and King, S. (**Mar. 2005**a), "Detection of Symbolic Gestural Events in Articulatory Data for Use in Structural Representations of Continuous Speech," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-05)* (IEEE Signal Processing Society Press, Philadelphia, USA), vol. I, pp. 885–888.

Gutkin, A. and King, S. (**May 24–25 2005**b), "Inductive String Template-Based Learning of Spoken Language," in *Proc. 5th International Workshop on Pattern Recognition in Information Systems (PRIS-2005), In conjunction with the 7th International Conference on Enterprise Information Systems (ICEIS-2005)*, edited by H. Gamboa and A. Fred (INSTICC Press, Miami, USA), pp. 43–51.

Haasdonk, B. (**Oct. 2003**), "Feature Space Interpretation of SVMs with non Positive Definite Kernels," Tech. Rep. 1/03, IIF-LMB, University of Freiburg, Germany.

Haasdonk, B. and Bahlmann, C. (**Sep. 2004**), "Learning with Distance Substitution Kernels," in *Proc. 26th German Association for Pattern Recognition (DAGM) Symposium* (Tübingen, Germany), pp. 220–227.

Halberstadt, A. K. and Glass, J. R. (**Sep. 1997**), "Heterogeneous Acoustic Measurements for Phonetic Classification," in *Proc. 5th European Conference on Speech Communication and Technology (Eurospeech'97)* (Rhodes, Greece), pp. 401–404.

Harris, J. (**1994**), *English Sound Structure* (Blackwell, Oxford).

Hazen, T. J., Hetherington, I. L., Shu, H., and Livescu, K. (**2005**), "Pronunciation modeling using a finite-state transducer representation," Speech Communication **46**, 189–203.

Hérault, J., Guérine-Dugué, A., and Villemain, P. (**Apr. 2002**), "Searching for the embedded manifolds in high-dimensional data, problems and unsolved questions,"

in *Proc. European Symposium on Artificial Neural Networks (ESANN'02)* (Bruges, Belgium), pp. 173–184.

Hjaltason, G. R. and Samet, H. (**May 2003**), "Properties of Embedding Methods for Similarity Searching in Metric Spaces," IEEE Trans. Pattern Analysis and Machine Intelligence **25**(5), 530–549.

Holmes, W. J. and Russell, M. J. (**1998**), "Probabilistic-trajectory Segmental HMMs," Computer Speech and Language **1**(13), 3–38.

Jacoby, S. L. S., Kowalik, J. S., and Pizzo, J. T. (**1972**), *Iterative Methods for Nonlinear Optimisation Problems* (Prentice-Hall, New Jersey).

Jain, A. K. and Dubes, R. C. (**1988**), *Algorithms for Clustering Data* (Prentice-Hall, New York).

Jain, A. K., Duin, R. P. W., and Mao, J. (**2000**), "Statistical Pattern Recognition: A Review," IEEE Trans. Pattern Analysis and Machine Intelligence **22**(1), 4–37.

Jain, A. K., Murty, M. N., and Flynn, P. J. (**1999**), "Data Clustering: A Review," ACM Computing Surveys **31**(3), 264–323.

Jakobson, R. (**1978**), *Six Lectures on Sound and Meaning* (The Harvester Press, Sussex, UK).

Jakobson, R., Fant, G. M., and Halle, M. (**1963**), *Preliminaries to Speech Analysis* (MIT Press, Cambridge, MA).

Jakobson, R. and Halle, M. (**1971**), *Fundamentals of Language* (Mouton de Gruyter, New York).

Jelinek, F. (**1985**), "Markov source modelling of text generation," in *The Impact of Processing Techniques on Communication*, edited by J. K. Skwirzynski (Martinus-Nijhoff Publishers, Dordrecht), pp. 569–598.

Jelinek, F. (**Mar. 1997**), *Statistical Methods for Speech Recognition* (MIT Press, Cambridge, MA).

Jiang, X., Muenger, A., and Bunke, H. (**2000**), "Computing the generalized mean of a set of graphs," in *Proc. 2nd Workshop on Graph-Based Representations (GbR'99) (collocated with Int. Conf. on Advances in Pattern Recognition (IAPR'00)* (Austrian Computer Society, Austria), pp. 115–124.

Juan, A. and Vidal, E. (**Aug. 2000**a), "Comparison of Four Initialization Techniques for the K-Medians Clustering Algorithm," in *Advances in Pattern Recognition: Joint IAPR International Workshops, SSPR 2000 and SPR 2000* (Springer, Alicante), vol. 1876, pp. 842–852.

Juan, A. and Vidal, E. (**Sep. 2000**b), "On the Use of Normalized Edit Distances and an Efficient k-NN Search Technique (k-AESA) for Fast and Accurate String Classification," in *Proc. 15th International Conference on Pattern Recognition (ICPR-2000)* (IEEE Computer Society Press, Barcelona, Spain), vol. 2, pp. 680–683.

Juan, A., Vidal, E., and Aibar, P. (**Aug. 1998**), "Fast k-Nearest-Neighbours searching through extended versions of the Approximating and Eliminating Search Algorithm (AESA)." in *Proc. 14th International Conference on Pattern Recognition (ICPR-98)* (IEEE Computer Society Press, Brisbane, Australia), vol. 1, pp. 828–830.

Jung, T.-P. (**1993**), "An Algorithm for Deriving an Articulatory-Phonetic Representation," Ph.D. thesis, Ohio State University.

Jung, T.-P., Krishnamurthy, A. K., Ahalt, S. C., Beckman, M. E., and Lee, S.-H. (**Jan. 1996**), "Deriving Gestural Scores from Articulatory-Movement Records Using Weighted Temporal Decomposition," IEEE Trans. Speech and Audio Processing **4**(1), 2–18.

Kamat, V. N. (**1995**), "Inductive Learning with the Evolving Tree Transformation System," Ph.D. thesis, Faculty of Computer Science, University of New Brunswick, Canada.

Kanal, L. N. (**1993**), "On patterns, categories, and alternate realities," Pattern Recognition Letters **14**, 241–255.

Kanungo, T., Mount, D. M., Netanyahu, N., Piatko, C., Silverman, R., and Wu, A. Y. (**2002**), "An Efficient k-Means Clustering Algorithm: Analysis and Implementation," IEEE Trans. Pattern Analysis and Machine Intelligence **24**(7), 881–892.

Kaplan, H. M. (**1971**), *Anatomy and Physiology of Speech* (McGraw-Hill), 2nd ed.

Kelley, C. T. (**1999**), "Detection and Remediation of Stagnation in the Nelder–Mead Algorithm Using a Sufficient Decrease Condition," Society for Industrial and Applied Mathematics Journal on Optimization **10**(1), 43–55.

Khamsi, M. A. and Kirk, W. A. (**2001**), *An Introduction to Metric Spaces and Fixed Point Theory*, Pure and Applied Mathematics: A Wiley-Interscience Series of Texts, Monographs, and Tracts (John Wiley & Sons, New York).

King, S. and Taylor, P. (**2000**), "Detection of Phonological Features in Continuous Speech using Neural Networks," Computer Speech and Language **14**(4), 333–353.

King, S., Taylor, P., Frankel, J., and Richmond, K. (**2000**), "Speech recognition via phonetically-featured syllables," in *PHONUS 5: Proc. Workshop on Phonetics and Phonology in ASR* (Saarbrücken, Institute of Phonetics, University of Saarland), pp. 15–34.

Kohonen, T. (**1985**), "Median Strings," Pattern Recognition Letters **3**, 309–313.

Kondrak, G. (**Apr. 2000**), "A New Algorithm for the Alignment of Phonetic Sequences," in *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-2000)* (Seattle), pp. 288–295.

Korkin, D. (**2003**), "A New Model for Molecular Representation and Classification: Formal Approach Based on the ETS Framework," Ph.D. thesis, Faculty of Computer Science, University of New Brunswick, Canada.

Kulkarni, S. R. and Lugosi, G. (**Oct. 1998**), "Learning Pattern Classification — A Survey," IEEE Trans. Information Theory **44**(6), 2178–2207.

Ladefoged, P. (**2001**), *A Course in Phonetics* (Harcourt Brace Jovanovich), 4th ed.

Lagarias, J. C., Reeds, J. A., Wright, M. H., and Wright, P. E. (**1998**), "Convergence Properties of the Nelder-Mead Simplex Method in Low Dimensions," Society for Industrial and Applied Mathematics Journal on Optimization **9**(1), 112–147.

Landau, E. (**1951**), *Foundations of Analysis* (Chelsea Publishing Co, New York).

Landau, G. M. and Vishkin, U. (**1986**), "Introducing efficient parallelism into approximate string matching and a new serial algorithm," in *Proc. 18th Symposium on Theory of Computing*, pp. 220–230.

Lange, L. H. (**1968**), *Elementary Linear Algebra* (John Wiley & Sons, New York).

Laub, A. J. (**2004**), *Matrix Analysis for Scientists and Engineers* (Society for Industrial and Applied Mathematics (SIAM)), chapter 4.

Laub, J. and Müller, K.-R. (**2004**), "Feature Discovery in Non-Metric Pairwise Data," Journal of Machine Learning Research **5**, 801–818.

Ledley, R. S., Belson, L. S., Jacobsen, M., Wilson, J., and Golab, T. (**1966**), "Pattern recognition studies in the biomedical sciences," in *Proc. AFIPS Spring Joint Comput. Conf* (Spartan Books, New York), vol. 28, pp. 411–430.

Lee, K.-F. and Hon, H.-W. (**Nov. 1989**), "Speaker-Independent Phone Recognition using Hidden Markov Models," IEEE Trans. Acoustics, Speech, and Signal Processing **37**(11), 1641–1648.

Levenshtein, V. I. (**1966**), "Binary codes capable of correcting deletions, insertions, and reversals," Cybernetics and Control Theory **10**(8), 707–710.

Liberman, A. M. and Mattingly, I. G. (**1985**), "The motor theory of speech perception revisited," Cognition **21**, 1–36.

Livescu, K., Glass, J., and Bilmes, J. (**Sep. 2003**), "Hidden Feature Models for Speech Recognition Using Dynamic Bayesian Networks," in *Proc. 8th European Conference on Speech Communication and Technology (Eurospeech-2003)* (Geneva, Switzerland), pp. 2529–2532.

Lui, Z.-Q., Cai, J., and Buse, R. (**2003**), *Handwriting Recognition*, vol. 133 of *Studies in Fuziness and Soft Computing* (Springer-Verlag, Berlin).

Mäkinen, V., Navarro, G., and Ukkonen, E. (**2003**), "Approximate Matching of Run-Length Compressed Strings," Algorithmica **35**, 347–369.

Manning, C. D. and Schütze, H. (**1999**), *Foundations of Statistical Natural Language Processing* (MIT Press, Cambridge, MA).

Martínez-Hinarejos, C. D., Juan, A., and Casacuberta, F. (**2003**), "Median strings for k-nearest neighbour classification," Pattern Recognition Letters **24**, 173–181.

Marzal, A. and Vidal, E. (**Sep. 1993**), "Computation of Normalized Edit Distance and Applications," IEEE Trans. Pattern Analysis and Machine Intelligence **15**(9), 926–932.

Masek, W. J. and Patterson, M. S. (**1980**), "A faster algorithm for computing string-edit distances," Journal of Computer and System Sciences **20**(1), 18–31.

Masek, W. J. and Patterson, M. S. (**1983**), "How to compute string-edit distances quickly," in *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, edited by D. Sankoff and J. B. Kruskal (Addison-Wesley, Reading, MA), pp. 337–349.

McKinnon, K. I. M. (**1998**), "Convergence of the Nelder–Mead Simplex method to a Nonstationary Point," Society for Industrial and Applied Mathematics Journal on Optimization **9**(1), 148–158.

Mitchel, T. M. (**1997**), *Machine Learning* (McGraw-Hill, New York).

Mohri, M. (**1997**), "Finite-State Transducers in Language and Speech Processing," Computational Linguistics **23**(2), 269–312.

Mohri, M., Pereira, F., and Riley, M. (**2002**), "Weighted finite-state transducers in speech recognition," Computer Speech and Language **16**(1), 69–88.

Mottl, V. V., Seredin, O. S., Dvoenko, S. D., Kulikowski, C. A., and Muchnik, I. B. (**Aug. 2002**), "Featureless Pattern Recognition in an Imaginary Hilbert Space," in *Proc. 16th International Conference on Pattern Recognition (ICPR'02)* (IEEE Computer Society Press, Québec City, Canada), vol. II, pp. 88–91.

Myers, E. W. (**1986**), "An $O(ND)$ difference algorithm and its variations," Algorithmica **1**, 251–256.

Needleman, S. B. and Wunsch, C. D. (**March 1970**), "A general method applicable to the search for similarities in the amino acid sequence of two proteins," Journal of Molecular Biology **48**(3), 443–453.

Nelder, J. A. and Mead, R. (**1965**), "A Simplex Method for Function Minimization," The Computer Journal **7**, 308–313.

Nguyen, N. (**2000**), "A Matlab toolbox for the analysis of articulatory data in the production of speech," Behaviour Research Methods, Instruments and Computers **32**, 464–467.

Nicolas, F. and Rivals, E. (**Jun. 2003**), "Complexities of the Centre and Median String Problems," in *Proc. 14th Symposium on Combinatorial Pattern Matching (CPM-03)*, edited by M. C. R. Baeza-Yates, E. Chávez (Springer-Verlag, Morelia, Mexico), vol. 2676 of *Lecture Notes in Computer Science*, pp. 315–327.

Nock, H. (**2001**), "Techniques for Modelling Phonological Processes in Automatic Speech Recognition," Ph.D. thesis, University of Cambridge, UK.

Olszewski, R. T. (**Feb. 2002**), "Generalized Feature Extraction for Structural Pattern Recognition in Time-Series Data," Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA.

Oommen, B. J. and Loke, R. K. S. (**1999**), "Designing Syntactic Pattern Classifiers using Vector Quantization and Parametric String Editing," IEEE Transactions on Systems, Man, and Cybernetics, Part B **29**(6), 881–888.

Ostendorf, M. (**Dec. 1999**), "Moving beyond the 'beads-on-a-string' model of speech," in *Proc. IEEE Automatic Speech Recognition and Understanding (ASRU'99) Workshop* (Keystone, USA).

Ostendorf, M., Digilakis, V., and Kimball, O. (**1996**), "From HMMs to Segment Models: A Unified View of Stochastic Modeling for Speech Recognition," IEEE Trans. Speech and Audio Processing **5**(4), 360–378.

Parkes, A. P. (**2002**), *Introduction to Languages, Machines and Logic: Computable Languages, Abstract Machines and Formal Logic* (Springer-Verlag, London).

Pavlidis, T. (**2003**), "36 years on the pattern recognition front," Pattern Recognition Letters **24**, 1–7.

Pękalska, E. (**Jan. 2005**), "Dissimilarity Representations in Pattern Recognition. Concepts, Theory and Applications," Ph.D. thesis, Technical University of Delft, Nederlands.

Pękalska, E. and Duin, R. P. W. (**Aug. 2002**), "Prototype selection for finding efficient representations of dissimilarity data," in *Proc. 16th International Conference on Pattern Recognition (ICPR'02)* (IEEE Computer Society Press, Québec City, Canada), vol. III, pp. 37–40.

Pękalska, E., Duin, R. P. W., Gunter, S., and Bunke, H. (**2004**), "On not making dissimilarities Euclidean," in *Joint IAPR International Workshops on Structural, Syntactic, and Statistical Pattern Recognition*, edited by A. Fred, T. Caelli, R. P. W. Duin, A. Campilho, and D. de Ridder (Springer-Verlag), vol. 3138 of *Lecture Notes in Computer Science*, pp. 1143–1152.

Pękalska, E., Paclík, P., and Duin, R. P. W. (**2002**), "A Generalised Kernel Approach to Dissimilarity based Classification," Journal of Machine Learning Research, Special Issue on Kernel Methods **2**(2), 175–211.

Perkell, J. S. (**1969**), *Physiology of Speech Production: Results and Implications of a Quantitative Cineradiographic Study*, Research Monograph No. 53 (MIT Press).

Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T. (**1986**), *Numerical Recipes: The Art of Scientific Computing* (Cambridge University Press, Cambridge, UK).

Pyenson, L. (**1977**), "Hermann Minkowski and Einstein's Special Theory of Relativity," Archive for History of Exact Science **17**(1), 71–95.

Richmond, K. (**1999**), "Estimating Velum Height from Acoustics during Continuous Speech," in *Proc. 6th European Conference on Speech Communication and Technology (Eurospeech'99)* (Budapest, Hungary), vol. 1, pp. 149–152.

Richmond, K. (**2001**), "Estimating Articulatory Parameters from the Acoustic Speech Signal," Ph.D. thesis, University of Edinburgh, UK.

Ripley, B. D. (**1996**), *Pattern Recognition and Neural Networks* (Cambridge University Press, Cambridge, UK).

Roth, V., Laub, J., Buhmann, J. M., and Müller, K.-R. (**2003**), "Going Metric: Denoising Pairwise Data," in *Advances in Neural Information Processing Systems (NIPS'15)*, edited by S. T. S. Becker and K. Obermayer (MIT Press, Cambridge, MA), pp. 817–824.

Rowe, E. G. P. (**2001**), *Geometrical Physics in Minkowski Spacetime*, Springer Monographs in Mathematics (Springer-Verlag, London).

Russell, M. J. and Bilmes, J. A. (**2003**), "Introduction to the special issue on new computational paradigms for acoustic modeling in speech recognition," Computer Speech and Language **17**(2-3), 107–112, (editorial).

Salomon, J., King, S., and Osborne, M. (**2002**), "Framewise Phone Classification using Support Vector Machines," in *Proc. 7th International Conference on Spoken Language Processing (ICSLP-2002)* (Denver, USA).

Sankoff, D. and Kruskal, J. B. (**1983**), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison* (Addison-Wesley, Reading, MA).

Schölkopf, B. and Smola, A. J. (**2001**), *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (MIT Press, New York).

Schrödinger, E. (**2003**), "Mind and Matter," in *What is Life? with* Mind and Matter *and* Autobiographical Scetches (Cambridge University Press, Cambridge, UK), 10th ed., pp. 90–164.

Sexl, R. U. and Urbantke, H. K. (**2001**), *Relativity, Groups, Particles: Special Relativity and Relativistic Symmetry in Field and Particle Physics*, Springer Physics (Springer-Verlag, Wien).

Shu, H. and Hetherington, L. I. (**2002**), "EM training of finite-state transducers and its application to pronunciation modelling," in *Proc. 7th International Conference on Spoken Language Processing (ICSLP-2002)* (Denver, USA), pp. 1293–1296.

Smith, T. F. and Waterman, M. S. (**1981**), "Identification of Common Molecular Subsequences," Journal of Molecular Biology **147**, 195–197.

Stender, J. and Addis, T. (Eds.) (**1990**), *Symbols against Neurons* (IOS Press, Amsterdam).

Stevens, K. (**1989**), "On the quantal nature of speech," Journal of Phonetics **17**, 3–46.

Ström, N. (**1996**), "The NICO Toolkit for Artificial Neural Networks," `http://www.speech.kth.se/NICO`.

Studdert-Kennedy, M. and Goldstein, L. M. (**2003**), "Launching language: The gestural origin of discrete infinity," in *Language Evolution*, edited by M. H. Christiansen and S. Kirby (Oxford University Press, Oxford), Studies in the Evolution of Language, chap. 13.

Sudkamp, T. (**1997**), *Languages and Machines* (Addison-Wesley).

Talkin, D. (**1995**), "Robust Algorithm for Pitch Tracking," in *Speech Coding and Synthesis*, edited by W. B. Kleijn and K. K. Paliwal (Elsevier Science), chap. 14, pp. 495–518.

Tanaka, E. (**1995**), "Theoretical Aspects of Syntactic Pattern Recognition," Pattern Recognition **28**(7), 1053–1061.

Taylor, P. and Black, A. (**1999**), "Speech synthesis by phonological structure matching," in *Proc. 6th European Conference on Speech Communication and Technology (Eurospeech'99)* (Budapest, Hungary), pp. 623–626.

Taylor, P., Black, A., and Cayley, R. (**2001**), "Heterogeneous relation graphs as a formalism for representing linguistic information," Speech Communication **33**, 153–174.

Tou, J. T. and Gonzalez, R. C. (**1974**), *Pattern Recognition Principles* (Addison-Wesley, New York).

Trubetskoy, N. S. (**1958**), *Grundzüge der Phonologie* (Vandenhoeck and Ruprech, Göttingen).

Valiente, G. (**2002**), *Algorithms on Trees and Graphs* (Springer-Verlag, Berlin).

Vapnik, V. (**1998**), *Statistical Learning Theory* (John Wiley & Sons, New York).

Vidal, E., Marzal, A., and Aibar, P. (**Sep. 1995**), "Fast Computation of Normalized Edit Distances," IEEE Trans. Pattern Analysis and Machine Intelligence **17**(9), 899–902.

Wagner, R. A. and Fisher, M. J. (**1974**), "The String-to-string correction problem," Journal of the ACM **21**(1), 168–173.

Walters, F. H., Parker, L. R., Morgan, S. L., and Deming, S. N. (**1991**), *Sequential Simplex Optimization* (CRC Press, Florida).

Watanabe, S. (**1985**), *Pattern Recognition: Human and Mechanical* (John Wiley & Sons, New York).

Wester, M. (**Sep. 2003**), "Syllable classification using articulatory acoustic features," in *Proc. 8th European Conference on Speech Communication and Technology (Eurospeech-2003)* (Geneva, Switzerland), pp. 233–236.

Wrench, A. A. (**2000**), "A multichannel articulatory database for continuous speech recognition research," in *Phonus5, Proceedings of Workshop on Phonetics and Phonology in ASR* (University of Saarland), pp. 1–13.

Wrench, A. A. and Hardcastle, W. J. (**2000**), "A multichannel articulatory database and its application for automatic speech recognition," in *Proc. 5th Seminar on Speech Production: Models and Data*, pp. 305–308.

Young, S. (**Mar. 1992**), "The General Use of Tying in Phoneme-Based HMM Speech Recognisers," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-92)* (IEEE Signal Processing Society Press, San Francisco), pp. 569–572.

Young, S. (**Aug. 2001**), "Statistical Modelling in Continous Speech Recognition (CSR)," in *Proc. 17th Int. Conference on Uncertainty in Artificial Intelligence* (Seattle, WA), pp. 562–571.

Zahorian, S. A., Silsbee, P. L., and Wang, X. (**1997**), "Phone Classification with Segmental Features and a Binary-Pair Partitioned Neural Network Classifier," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-97)*, vol. 2, pp. 1011–1014.

Zemlin, W. R. (**1968**), *Speech and Hearing Science: Anatomy and Physiology* (Prentice-Hall, New Jersey).

Zweig, G., Bilmes, J., Richardson, T., Filali, K., Livescu, K., Xu, P., Jackson, K., Brandman, Y., Sandness, E., Holtz, E., Torres, J., and Byrne, B. (**Jun. 2002**), "Structurally Discriminative Graphical Models for Automatic Speech Recognition:

Results from the 2001 Johns Hopkins Summer Workshop," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-02)* (IEEE Signal Processing Society Press, Orlando, FL), vol. I, pp. 93–96.